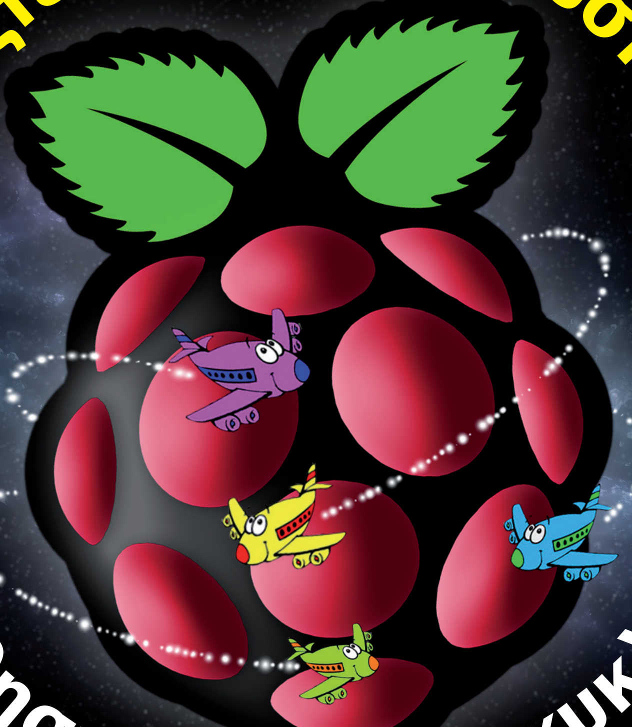
all rights reversed

HACKER

Αύγουστος 2013 # Τιμή τεύχους 4€

τεύχος 023 - Die Leckere Käsetaschen Edition

Φτιάξτε το δικό σας σύστημα παρακολούθησης εναέριας κυκλοφορίας [*]



Ακονίστε τη λεπίδα της ZTE

Παντρέψτε τα Windows με το OpenBSD

Ελέγξτε τον υπολογιστή σας, μέσω του Dropbox

[*]...όχι κατ' ανάγκη με το Raspberry Pi :)

all rights reversed

HACKER

tefhos 23o

Περιεχόμενα

και οδηγίες χρήσης

Εκδότης / Διευθυντής	Χρήστος Βαρελάς
Αρχισυντάκτης	Παναγιώτης Βαρελάς
Σύμβουλοι έκδοσης	Νίκος Μουρατίδης, Βούλα Παυλίδου
Σύνταξη	Παναγιώτης Βαρελάς, Χρήστος Βαρελάς, Ανδρέας Βενιέρης, Γιώργος Γιάννου, Μανώλης Κιαγιάς, Ιωάννης Κονιάρης, Πέτρος Κυλαδίτης, Γιώργος Μελέκος, Νίκος Μουρατίδης, Νικόλαος Σηφάκης, Μέσος Παπαδόπουλος (TM), Χρίστος Τόμπρας
Γραμματική επιμέλεια	Βούλα Παυλίδου
Καλλιτεχνική Επιμέλεια	Πέτρος Φιλιππίδης

Το ηλεκτρονικό περιοδικό deltaHacker είναι συνδρομητικό και εκδίδεται από την Parabing Creations δώδεκα φορές το χρόνο, κάθε μήνα. Η Parabing Creations έχει την έδρα της στη Μαιώτιδος 3, 55133, Καλαμαριά. Ο δικτυακός τόπος του περιοδικού είναι στο http://deltahacker.gr. Το email επικοινωνίας είναι το talk2us@deltahacker.gr και ειδικά για τις συνδρομές είναι το subscriptions@deltahacker.gr. Οι γενικές πληροφορίες για τις συνδρομές παρατίθενται στο http://deltahacker.gr/subscriptions και οι ίδιες οι παραγγελίες γίνονται από το http://deltahacker.gr/pdf12order. Όλες οι απόψεις που εκφράζονται στα άρθρα δεν εκφράζουν απαραίτητα και τη γνώμη του περιοδικού.

(*) Βλέπετε deltaCast. Δεν είναι και τόσο κακό.
http://deltahacker.gr/category/deltacast

Αόρατοι στο Internet ...ή και όχι

Η βροχή αποκαλύψεων των τελευταίων εβδομάδων το δείχνει ξεκάθαρα: Ό,τι κάνουμε στο Διαδίκτυο *οφείλουμε* να θεωρούμε πως γίνεται δημόσια, διαφορετικά κοροϊδεύουμε εαυτούς. Αξίζει όμως να επιδιώκουμε την απόλυτη ανωνυμία ή/και privacy στον online κόσμο;

by subZraw

Αν θεωρείτε ότι η τελευταία ερώτηση επιδέχεται σαφέστατα καταφατική απάντηση και μάλιστα δεν θα έπρεπε καν να έχει τεθεί, σταματήστε εδώ το διάβασμα. Δείτε, αν είναι, το άρθρο για την κατασκευή ενός συστήματος παρακολούθησης της εναέριας κυκλοφορίας (σελ. 12), διαβάστε για τις μάταιες προσπάθειες που έκαναν οι Γερμανοί προκειμένου να ισχυροποιήσουν την κρυπτογράφηση της Enigma (σελ. 36), ή μάθετε πώς μπορείτε να πατρέψετε τα Windows με το OpenBSD (σελ. 62).

Προς όλους τους υπόλοιπους, κατ' αρχάς απολογίες για το περιττό διαφημιστικό που μόλις διαβάσατε. Κατά δεύτερον, να ξεκαθαρίσουμε ευθύς εξαρχής ότι, σύμφωνα με τη δική μας κοσμοαντίληψη, η πιθανότητα να παραβιάζεται η ανωνυμία ή/και το privacy ενός οποιουδήποτε χρήστη στο Internet είναι, πρακτικά, μηδέν. Ας το διατυπώσουμε κι αλλιώς: Κατά πάσα πιθανότητα, κανείς δεν ασχολείται μαζί σας αυτή τη στιγμή. Δεν υπάρχει άνθρωπος στον πλανήτη που να ενδιαφέρεται για το τι (δεν) κάνετε στο Internet — και μάλλον δεν πρόκειται ποτέ να βρεθεί κάποιος που θα ενδιαφερθεί.

Πραγματικά το πιστεύουμε αυτό, ωστόσο εδώ υπάρχουν δύο ύπουλα σημεία. Το ένα έχει να κάνει με το ότι η προηγούμενη εκτίμηση αφορά σε όλον τον πληθυσμό κι όχι σε καθένα από τα μέλη του. Είναι σαν τον μέσο όρο, ο οποίος ναι μεν δίνει μια γενική εικόνα αλλά κρύβει τις εκπλήξεις. Οι ελληνίδες, για παράδειγμα, έχουν μέσο ύψος 1,66μ[1], ωστόσο αυτό δεν σημαίνει πως δεν υπάρχουν ελληνίδες με ύψος 1,78μ. Έτσι και με τις παρακολουθήσεις στο Internet: Το ότι κανείς δεν ενδιαφέρεται, π.χ., για την ηλεκτρονική σας αλληλογραφία, δεν σημαίνει ότι αυτή τη στιγμή δεν υπάρχουν ορισμένοι τύποι που ψάχνουν το inbox άλλων χρηστών — εν αγνοία τους φυσικά.

Το άλλο ύπουλο σημείο έχει περισσότερο ηθική διάσταση. Κάθεται λοιπόν ένας χρήστης και το σκέφτεται από εδώ, το εξετάζει από εκεί και καταλήγει στο συμπέρασμα ότι δεν έχει φοβερά ή σπουδαία μυστικά να κρύψει, οπότε δεν υπάρχει κανένας λόγος για προβληματισμό ή ανησυχία. Μετά από λίγο, όμως, αρχίζει να ενοχλείται. Η σκέψη ότι κάποιος άλλος μπορεί να μάθει πράγματα για εκείνον χωρίς τη συγκατάθεσή του, δεν είναι και η πιο ευχάριστη. Εντάξει, κακούργος δεν είναι, ληστεία ή τρομοκρατική ενέργεια δεν σχεδιάζει κ.ο.κ., από πότε όμως καταργήθηκε το *δικαίωμα* στην ανωνυμία, ειδικά στο Διαδίκτυο; Η αλληλογραφία και γενικά η ηλεκτρονική του επικοινωνία, από πότε είναι διαθέσιμη στον οποιονδήποτε και γιατί δεν έχει ενημερωθεί γι' αυτό; Γενικά, έχουμε ή δεν έχουμε το δικαίωμα να επιλέγουμε το αν θα είμαστε ανώνυμοι ή/και το αν θα προστατεύουμε την επικοινωνία μας από τρίτους;

Κοιτάξτε, μεταξύ της αφελούς στάσης του στιλ «σιγά, αυτά συμβαίνουν μόνο σε άλλους» και της παράνοιας του στιλ «μας παρακολουθούν όλους και διαρκώς»[2], ο γράφων ανέκαθεν διάλεγε μια θέση κάπου στη μέση. Αν μη τι άλλο, όταν κάποιος επιθυμεί —και οι

1. Wikipedia, http://en.wikipedia.org/wiki/Human_height#Average_height_around_the_world
2. ...και φυσικά μας ψεκάζουν!

λόγοι πραγματικά δεν έχουν σημασία— να ‵ναι ανώνυμος ή/και να κρυπτογραφεί ισχυρά την επικοινωνία του, τότε πρέπει να ‵ναι σε θέση να το κάνει! Βεβαίως, αυτό προϋποθέτει ελεύθερη κι ανεμπόδιστη πρόσβαση στα κατάλληλα εργαλεία λογισμικού. Ας πούμε, λοιπόν, ότι επιλέγω να είμαι ανώνυμος ή/και να κρυπτογραφώ την επικοινωνία μου. Είναι εφικτό να πετυχαίνω τους δύο αυτούς στόχους; Φυσικά! Είναι εύκολο; Χμ, έτσι κι έτσι.

Ας δούμε λίγο το θέμα της ανωνυμίας. Για το Tor σίγουρα θα έχετε ακούσει. Στο περιοδικό έχουμε γράψει αρκετές φορές για το δημοφιλέστατο δίκτυο ανωνυμίας. Στο τεύχος 022, π.χ., δείχνουμε πώς είναι δυνατόν να περνάμε όλη τη δικτυακή κίνηση ενός ολόκληρου LAN μέσα από το Tor. Αλλά ξέρετε κάτι; Η καθημερινή και διαρκής χρήση του Tor network δεν είναι ό,τι πιο πρακτικό, αφού συχνά το surfing μέσω Tor είναι αισθητά αργό. Και κάπως έτσι αρχίζει να εκδηλώνεται —για άλλη μια φορά— η διαρκής ένταση μεταξύ ευκολίας κι ασφάλειας...

Πάμε να δούμε και το θέμα του privacy, εξετάζοντάς το από τη σκοπιά του ανθρώπου που θέλει να ‵ναι βέβαιος ότι την αλληλογραφία του τη διαβάζουν οι παραλήπτες που πρέπει και κανένας άλλος. Ας πούμε ότι τα emails αποστέλλονται μέσω μιας υπηρεσίας web σαν αυτές των Gmail, Yahoo! κ.ά. Ο χρήστης μας γνωρίζει αρκετά, ώστε να μη διαχειρίζεται την αλληλογραφία του μέσα από επισφαλή κανάλια. Ελέγχει, συγκεκριμένα, το αν η σύνδεση με τον απομακρυσμένο server είναι HTTPS, κι αν δεν είναι πολύ απλά δεν προχωρά. Έχει βέβαια κατά νου ότι το HTTPS δεν αποτελεί πανάκεια3, ωστόσο είναι σε θέση να αξιολογεί το αν υπάρχει πιθανότητα —αλλά και δυνατότητα— για MiTM attack. Την ίδια στιγμή, ξέρει πολύ καλά ότι τα emails δεν αποθηκεύονται κρυπτογραφημένα στους απομακρυσμένους servers. Πράγματι, αν ήταν κρυπτογραφημένα τότε τα φίλτρα anti-SPAM πολύ απλά δεν θα δούλευαν (αλλά δουλεύουν μια χαρά). Αποφασίζει, λοιπόν, να πάρει τα πράγματα στα χέρια του και να ασφαλίζει μόνος του το περιεχόμενο των emails, καταφεύγοντας σε τεχνικές κρυπτογραφίας δημοσίου κλειδιού (public key cryptography). Φτιάχνει, λοιπόν, ένα ζεύγος ιδιωτικού-δημοσίου κλειδιού και δημοσιεύει κάπου το δεύτερο, ώστε ο καθένας να μπορεί να το έχει. Τώρα, αν θέλει να στείλει σε κάποιον ένα κρυπτογραφημένο email, πρέπει πρώτα να τον πείσει ώστε να χρησιμοποιεί κι εκείνος το ίδιο σύστημα κρυπτογραφίας και φυσικά θα του ζητήσει το δημόσιο κλειδί του. Αφού το πάρει, θα κρυπτογραφήσει μ’ αυτό το περιεχόμενο του email, θα το στείλει στον παραλήπτη κι εκείνος θα το αποκρυπτογραφήσει με το αντίστοιχο ιδιωτικό κλειδί το οποίο μόνον αυτός διαθέτει. Αντίστροφα, αν πάει κάποιος να στείλει email στον φίλο μας, τότε εκείνος θα τον προτρέψει να πάρει το δικό του δημόσιο κλειδί, να κρυπτογραφήσει το περιεχόμενο και μετά να του στείλει email. Αφού το λάβει, θα αποκρυπτογραφήσει το περιεχόμενό του με το αντίστοιχο ιδιωτικό κλειδί, το οποίο μόνον εκείνος έχει. Δεν νομίζουμε ότι χρειάζεται ν’ αναφέρουμε περισσότερες λεπτομέρειες, προκειμένου να δείξουμε πόσο γρήγορα γίνεται άβολη η αποστολή και λήψη κρυπτογραφημένης αλληλογραφίας.

Συμπερασματικά, η ανωνυμία και το privacy αποτελούν θεμελιώδες ανθρώπινο δικαίωμα — κι ας λένε ό,τι θέλουν οι τοπικοί νόμοι. Πάντως η συντριπτική πλειονότητα των χρηστών (δεν εξαιρούμε εαυτούς) μάλλον δεν έχει νόημα να φροντίζει ανά πάσα στιγμή για τη διασφάλιση της ανωνυμίας ή/και του privacy. Όποτε όμως κάποιος επιλέγει να εξασκήσει το δικαίωμά του, προφανώς πρέπει να έχει πρόσβαση στις κατάλληλες τεχνολογίες κι εργαλεία. Χάρη στα Ανοικτά Πρότυπα και στο Ελεύθερο/Ανοικτό Λογισμικό, η πρόσβαση είναι πράγματι εφικτή. Όμως το πόσο πρακτική είναι η καθημερινή ή έστω συχνή χρήση τεχνικών διασφάλισης της ανωνυμίας και του privacy, είναι εντελώς διαφορετικό ζήτημα. Θα επανέλθουμε σύντομα, με λεπτομέρειες και πρακτικές συμβουλές.

3. Μάλλον θα έχει διαβάσει το άρθρο που ξεκινά από τη σελίδα 12 του deltaHacker 020 :)



Ουέλ, μέλ

Αγαπητοί μου,

Από τον έκτο όροφο του Royal Festival Hall που σας γράφω, τα πράγματα δείχνουν επαρκώς απομακρυσμένα, σχεδόν μικροσκοπικά, αν και λίγο γκρίζα.

Η μέρα ξεκίνησε συννεφιασμένη αλλά το δελτίο προβλέπει ότι το απόγευμα θα έχει λιακάδα.

Θα αρχίσει από τις τέσσερις και μετά.

Η λιακάδα εννοώ.

Πολύ πιθανόν, αλλά αυτή τη στιγμή που σας γράφω, γύρω στις 2 και τέταρτο το μεσημέρι, δεν μπορώ να διακρίνω καμία ένδειξη της επερχόμενης λιακάδας.

Από την άλλη, δεν μπορώ παρά να εμπιστευτώ το δελτίο καιρού, το οποίο παρακολουθώ στο άιΦον της Μαιρούλας όλες αυτές τις βδομάδες που έχουμε έρθει Λονδίνο.

Δεν μπορείς να κάνεις και αλλιώς, βλέπεις.

Οι Βρετανοί έχουν αναγάγει την έννοια του δελτίου καιρού σε βασικό νήμα του κοινωνικού ιστού. Θα σας είναι οικείο, βέβαια, το στερεότυπο, που θέλει τις Αγγλίδες γιαγιάδες, ντυμένες με το ροζ ή σωμόν ταγιέρ τους, να πίνουν afternoon tea με λίγο γάλα, να τρώνε butter cookies και να μιλάνε για τον καιρό.

Ο καιρός είναι το βασικό, το πάγιο θέμα συζήτησης.

Το σημαντικότερο, βέβαια, είναι πως μια τέτοια συζήτηση δεν έχει ως στόχο την ανταλλαγή πληροφοριών.

Μιλώντας γενικώς, ο καιρός, στην Αγγλία, είναι είτε συννεφιασμένος και μονότονος, είτε εξαιρετικά άστατος. Όταν είναι άστατος, την μία στιγμή μπορεί να βρέχει καταρρακτωδώς και μερικά λεπτά αργότερα να σε τυφλώνει ο ήλιος.

Σας το λέω από πείρας. Προ ημερών είχαμε κατέβει στην Όξφορντ Στρητ με την Μαιρούλα και την εξαδέλφη, και μας πιάνει η μπόρα στον δρόμο. Οι μπόρες --μπόρες εννοώ με την έννοια της δυνατής βροχής που σε κάνει μούσκεμα-- είναι σπάνιες στο Λονδίνο, οπότε ήμασταν απροετοίμαστοι. Μπαίνουμε τρέχοντας στο πρώτο μαγαζί που βλέπουμε μπροστά μας για να αγοράσουμε ομπρέλα. (Παρεμπιπτόντως, ομπρέλες μοιάζει σαν να πουλάνε όλα τα μαγαζιά, ακόμα και τα τυροπιττάδικα ή τα κοσμηματοπωλεία.)

Τέλος πάντων, να μην τα πολυλογώ, παίρνουμε ομπρέλες και σε λίγο ξαναβγαίνουμε πλήρως εξοπλισμένοι ακόμα και για μουσώνα. Και τι να δούμε;

Βροχή, γιοκ. Ο ήλιος ψηλά στον ουρανό, ο ουρανός καταγάλανος, χωρίς ένα συννεφάκι, και οι δρόμοι στεγνοί.

Μην με ρωτάτε πώς. Συνέβη. Το είδα με τα μάτια μου.

Είναι αδύνατο για οποιονδήποτε να κατανοήσει τον αγγλικό καιρό όταν είναι άστατος --πόσο μάλλον να τον προβλέψει-- και αυτό ισχύει τόσο για μας όσο και για τις Αγγλίδες γιαγιάδες.

Εκτός από την Μαιρούλα που είναι γενικώς χαλαρή, μιας και κυκλοφορεί με τα άιΦον και τα άιΠαντ υπο μάλης, και έχει ενημέρωση real time για τον καιρό των επομένων 24 ωρών, μοιρασμένη σε τρίωρα και συγκεκριμενοποιημένη για το ακριβές σημείο στο οποίο βρίσκεται ανά πάσα στιγμή (GPS γαρ).

Εξ ου και η πληροφορία πως σήμερα μετά τις 4 θα έχει λιακάδα στην ευρύτερη περιοχή του Royal Festival Hall.

Δεν μασάω. Το αναφέρω μόνο για την καινοτομία του πράγματος.

Κατά την γνώμη μου αυτά τα μετεωρολογικά γκατζετάκια χαλάνε κάτι από την γοητεία (και την έκπληξη) του να είσαι στην Αγγλία.

Φανταστείτε τώρα να είστε Αγγλίδα γιαγιά, να συζητάτε για τον καιρό πίνοντας afternoon tea με τις φιλενάδες σας, και να πετάγεται ξαφνικά η κάθε Μαιρούλα για να σας πει ότι σύμφωνα με το τελευταίο δελτίο καιρού, από τις τέσσερις μέχρι τις εφτά σήμερα το απόγευμα προβλέπεται ηλιοφάνεια στην περιοχή Ουέστμίνστερ / Εμπάνκμεντ, με τα σύννεφα να ξαναπυκνώνουν αργότερα, και με σποραδικές βροχές να αναμένονται μετά την δεκάτη βραδυνή.

Σπάσιμο. Το πιθανότερο είναι ότι θα την ακούγατε προσεκτικά και μετά, αμήχανα, θα αλλάζατε θέμα.

Η συζήτηση για τον καιρό είναι μια από εκείνες τις μικρές καθημερινές τελετουργίες της Αγγλικής κοινωνικής ζωής που επιβεβαιώνουν και επισφραγίζουν, στους συμμετέχοντες, την αίσθηση ταυτότητας και κοινότητας. Δεν συζητάς για να μάθεις. Συζητάς για να επιβεβαιώσεις ότι κάτι σε συνδέει με τον συνομιλητή σου.

Κάτι σαν την βασιλική οικογένεια, τον πρίγκηπα Ουίλιαμ, ή το βασιλικό βρέφος. Αν είσαι Άγγλος, έχεις γνώμη. Θετική ή αρνητική, αδιάφορο. Το σημαντικό είναι πως έχεις γνώμη. Και το γεγονός αυτό, η ύπαρξη της γνώμης δηλαδή, επιβεβαιώνει την αγγλοσύνη σου.

Τέλος πάντων, για να ξαναγυρίσω στα καθΥ ημάς, σήμερα έχουμε έρθει οικογενειακώς στο Royal Festival Hall και καθόμαστε στα τραπέζια του έκτου ορόφου. Γύρω μας όλοι έχουν από ένα λάπτοπ και κάτι κάνουνε. Μόνο εγώ είμαι η παραφωνία, με την γραφομηχανή μου, αλλά προσπαθώ να μην δίνω στόχο. Έχω κάτσει παράμερα και τους παρακολουθώ από απόσταση.

Με βοηθάει η απόσταση, πρέπει να το παραδεχτώ.

Προσπαθώ, βέβαια, να μην κάνω το λάθος που σας έγραφα την άλλη φορά, να μπερδεύω δηλαδή τα επίπεδα αναφοράς των γεγονότων στα οποία συμμετέχω.

Εγώ, το βασικό γεγονός στο οποίο συμμετέχω όλες αυτές τις βδομάδες είναι η επίσκεψή μας στο Λονδίνο για να βοηθήσουμε τάχα μου τον ανηψιό με την μετακόμισή του.



Χρησιμοποιώ λοιπόν τον όρο «απόσταση» (από την οποία παρακολουθώ τα καθέκαστα), υπό την χωρική του έννοια: Κάθομαι παράμερα. That's all.

Πλάκα πλάκα ήρθαμε Λονδίνο για μερικές μέρες κι έχουμε μείνει κοντά έναν μήνα τώρα. Όλο αγγλικά μιλάμε. Hello και please και thank you όλη την ώρα.

Η πατρίδα μεγαλουργεί, τι να σας λέω.

Την περασμένη βδομάδα ήρθανε να μας βρούνε η κυρία Καίτη και η μητέρα της εξαδέλφης. Τους έφαγαν τ' αφτιά με το σκάιπ η Μαιρούλα και η εξαδέλφη, και το πήρανε απόφαση να αφήσουνε για λίγο το Πόρτο Γερμενό και τα τζιτζίκια για να έρθουνε να δούνε το περιβόητο καινούργιο σπίτι του ανηψιού.

Βέβαια, η βασική ιδέα της μητέρας της εξαδέλφης, που είναι ινδουίστρια, ήταν να ξαναεπισκεφτεί την Μπρικ Λέιν για να αγοράσει (κι άλλα) παραδοσιακά σάρι.

Λέω «ξανά», γιατί και πέρσι που είχαμε έρθει για τους Ολυμπιακούς, εκεί μας έτρεχε.

Πήγανε, τελικά, από Μπρικ Λέιν (χωρίς εμάς) αλλά σάρι δεν αγοράσανε. Αγοράσανε όμως ένα σακουλάκι με καπνιστό τσίλι «φάντασμα», όπως το λένε --«νάγκα μπουτ τζολοκία» επί το ινδικότερον-- και είναι κατευχαριστημένες.

«Καλά, ήρθατε στο Λονδίνο να αγοράσετε τσίλι;» ρώτησε η Μαιρούλα.

«Μη μιλάς αν δεν γνωρίζεις», ήταν η απάντηση.

«Σκοπεύετε πράγματι να το χρησιμοποιήσετε;» ρώτησα κι εγώ με την σειρά μου.

«Γιατί, εσύ όταν αγοράζεις όλα αυτά τα βιβλία που αγοράζεις σκοπεύεις πράγματι να τα διαβάσεις;» μου απάντησε η μητέρα της εξαδέλφης αινιγματικά.

«Με αδικείτε!» της είπα.

«Εσύ με αδικείς, που αμφισβητείς τις προθέσεις μου. Φυσικά και θα το χρησιμοποιήσω.»

«Αχ, ωραία, μου αρέσουν τα καυτερά», είπε η Μαιρούλα.

«Χμμμ» είπα εγώ.

Τέλος πάντων. Γενικώς καλά περνάμε.

Βοηθάει βέβαια και ο καιρός, που φέτος μας επιφύλαξε τις καλύτερες εκπλήξεις. Μετά από τις πρώτες δέκα μέρες, που ήταν μια από τα ίδια, γκριζάδα και συννεφιά, έβγαλε ξαφνικά ήλιο κι έμεινε με τον ήλιο. Μέχρι χτες.

Χτες ξεκίνησε να ρίχνει όλη μέρα καρεκλοπόδαρα. Την νύχτα σταμάτησε, και σήμερα έχει συννεφιά. Όπως σας είπα, ήλιος αναμένεται μετά τις τέσσερις.

Το υπόλοιπο της εβδομάδας αναμένεται εξ ίσου καλοκαιρινό, αλλά εμείς δεν θα το δούμε. Επιστρέφουμε όλοι Τετάρτη βράδυ, δηλαδή αύριο. Όλοι σημαίνει όλοι. Και ο ανηψιός.

Ίσα ίσα για να παρακολουθήσουμε ιδίοις όμμασι την έλευση του Αυγούστου και την πτώση των τιμών στα φαγάδικα και εστιατόρια λόγω της μείωσης του ΦΠΑ στην εστίαση.

Κρατάμε την ανάσα μας, τι να σας λέω.

Και περιμένουμε βέβαια να δούμε τι θα μας φτιάξει η θεία με το μπουτ τζολοκία.

Θα σας ενημερώσω (αν επιζήσω).

Σας ασπάζομαι

Θείος Ακάκιος

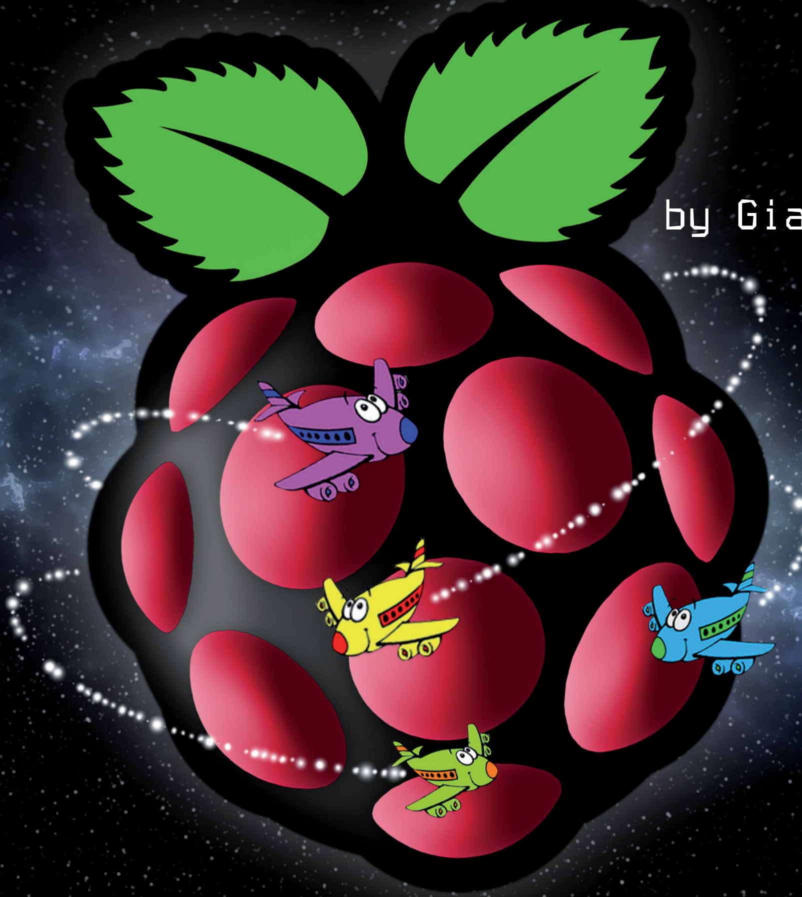


Πλήρη άρθρα που συμπληρώνουν το περιοδικό [**addenda**]. Το video podcast [**netcasting → deltaCast**]. Άρθρα που μας ήρθε να γράψουμε, έτσι, απλά [**/dev/random**]. Τα προκλητικά **mind hacks**. Τα βίντεο από τα προηγούμενα live streams [**netcasting → deltaStream**]. Ολόκληρα blasts from the past [**bftp**]. Τα πάντα ενδιαφέροντα **forums**. Το σημείο συνάντησης για τους φίλους του περιοδικού.

deltahacker.gr

Αληθινό, διασκεδαστικό hacking **για όλους**!

Skill: Intermediate
Tags: ADS-B, tuner, SDR, RTL2832U, RTLSDR, dump1090



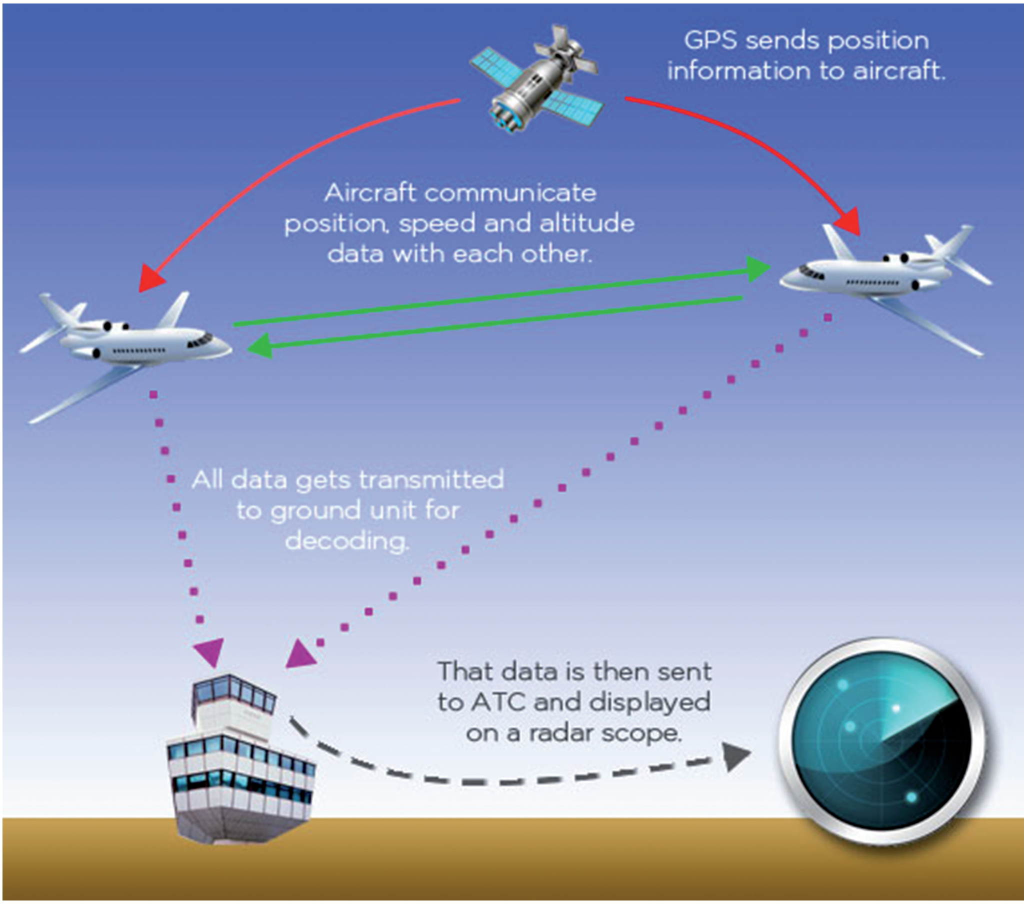
by GiannouG

Software-Defined Radio

Μια από τις καλοκαιρινές συνήθειες του αρθρογράφου (ειδικά όταν δεν έχει πρόσβαση στο Διαδίκτυο) είναι να κάθεται τα βράδια και να χαζεύει τον ουρανό. Συχνά-πυκνά, αρκεί βέβαια να φοράει τα γυαλιά μυωπίας του, εντοπίζει μερικά αεροπλάνα κι αμέσως βγάζει το κινητό για να μάθει ποια πτήση είναι, σε τι ύψος πετάει και με ποια ταχύτητα. Βλέπετε, υπάρχουν αρκετές εφαρμογές για smartphones που κάνουν αυτήν ακριβώς τη δουλειά. Οι εν λόγω εφαρμογές αντλούν τα δεδομένα από διάφορους χρήστες, οι οποίοι συλλέγουν και διανέμουν τις σχετικές πληροφορίες δωρεάν. Αναρωτιέστε πώς πληροφορούνται αυτοί οι χρήστες για τις πτήσεις;

Τα τελευταία χρόνια έχει αρχίσει να χρησιμοποιείται ένα νέο σύστημα ελέγχου των αεροσκαφών. Πρόκειται για το ADS-B (Automatic Dependent Surveillance-Broadcast). Από το 2020 και μετά, το συγκεκριμένο σύστημα θα είναι υποχρεωτικό για τα αεροσκάφη που πετάνε πάνω από τις ΗΠΑ, ενώ σε λίγα χρόνια θα επιβληθεί και στις πτήσεις πάνω από την Ευρώπη. Σήμερα χρησιμοποιείται ήδη από αρκετά αεροσκάφη, μεταξύ των οποίων και τα περισσότερα Airbus. Το μόνο σίγουρο είναι ότι όσο περνάει ο καιρός, τόσο περισσότερο θα εξαπλώνεται η χρήση του. Οι χρήστες που παρέχουν πληροφορίες για τις πτήσεις στις σχετικές εφαρμογές για smartphones, διαθέτουν ειδικούς δέκτες με τη βοήθεια των οποίων λαμβάνουν τα σήματα του εν λόγω συστήματος! Τι θα λέγατε αν γινόμασταν κι εμείς διανομείς τέτοιων πληροφοριών; Το μόνο που χρειαζόμαστε είναι ο κατάλληλος δέκτης. Μην νομίζετε ότι θα σας προτείνουμε να αγοράσετε κάποια εξειδικευμένη (κι ενδεχομένως πανάκριβη) συσκευή λήψης. Θα στηριχτούμε απλά στο λεγόμενο Software Defined Radio...

Το ADB-S δημιουργήθηκε ώστε οι πιλότοι να γνωρίζουν ανά πάσα στιγμή πόσα και ποια αεροπλάνα βρίσκονται κοντά τους, προς τα πού κατευθύνονται και σε ποιο υψόμετρο πετούν. Το ίδιο σύστημα χρησιμοποιείται φυσικά και από τους πύργους ελέγχου, για τη ρύθμιση της εναέριας κυκλοφορίας.



Η κεντρική ιδέα

Η βασική ιδέα πίσω από το Software Defined Radio είναι η εξής: Οι κάρτες TV tuner που μπορούμε να συνδέσουμε στον υπολογιστή μας, περιλαμβάνουν δέκτες που μπορούν να λαμβάνουν σήματα σε ένα ευρύ φάσμα συχνοτήτων και σίγουρα μεγαλύτερο από εκείνο που χρησιμοποιείται για τις τηλεοπτικές εκπομπές. Όμως, σε αρκετές συσκευές του είδους, το hardware στηρίζει πολλές από τις λειτουργίες του στους αντίστοιχους drivers (όπως συνέβαινε και στα win-modem, αν σας θυμίζει κάτι αυτό). Μεταξύ άλλων, το software που συνοδεύει τις συσκευές, οι drivers δηλαδή, περιορίζουν τη λήψη στις συχνότητες του τηλεοπτικού σήματος. Επομένως, αν θέλουμε να συντονιστούμε σε μια αυθαίρετα επιλεγμένη συχνότητα, αρκεί να *πειράξουμε* τον driver ή απλά να χρησιμοποιήσουμε μια ήδη τροποποιημένη εκδοχή του. Έτσι, με τη βοήθεια ενός TV tuner και του κατάλληλου driver, μπορούμε να λαμβάνουμε σχεδόν οποιοδήποτε σήμα!

Ας επιστρέψουμε τώρα στο θέμα μας. Τα σήματα του συστήματος ADS-B εκπέμπονται σε συχνότητες που λαμβάνονται άνετα από αρκετούς δέκτες ψηφιακού τηλεοπτικού σήματος. Έτσι, με τη βοήθεια ενός τέτοιου δέκτη κι ενός ειδικού driver, μπορούμε να λάβουμε τα σχετικά σήματα πανεύκολα. Στη συνέχεια, μπορούμε να αποκωδικοποιήσουμε τα πακέτα του ADS-B και να εξαγάγουμε όλες τις πληροφορίες που αναζητάμε. Τέλος, εφόσον το επιθυμούμε, μπορούμε να δημοσιεύσουμε τα ευρήματά μας μέσω κάποιας σχετικής υπηρεσίας.

Αυτός ο μικρός TV tuner προορίζεται για τις συχνότητες της τηλεόρασης και του ραδιοφώνου, αλλά είναι ικανός να συντονιστεί και σε πολλές άλλες!



Συλλογή υλικού

Πρώτα απ’ όλα θα χρειαστούμε έναν υπολογιστή με Linux! Εμείς αποφασίσαμε να χρησιμοποιήσουμε ένα Raspberry Pi, εξαιτίας της χαμηλής κατανάλωσης που το καθιστά ιδανικό για πολύωρη λήψη δεδομένων[1]. Εσείς, ωστόσο, μπορείτε να χρησιμοποιήσετε οποιοδήποτε σύστημα και με οποιαδήποτε διανομή. Στη συνέχεια πρέπει να επιλέξουμε και να προμηθευτούμε τον κατάλληλο TV tuner. Εδώ απαιτείται λίγη προσοχή! Κατ’ αρχάς, χρειαζόμαστε κάποιον TV tuner (εν είδει USB stick) με το ολοκληρωμένο κύκλωμα Realtek RTL2832U. Μια αναζήτηση στο eBay θα εμφανίσει πολλά προϊόντα, με τιμές που κυμαίνονται περί τα 13 ευρώ. Τώρα, πρέπει να σταθούμε σε μία ακόμα λεπτομέρεια. Τα USB sticks που χρησιμοποιούν το συγκεκριμένο ολοκληρωμένο, διαθέτουν ένα ακόμα τσιπάκι, που είναι υπεύθυνο για το συντονισμό στις διάφορες συχνότητες. Όμως αυτό το τσιπ δεν είναι ίδιο για όλα τα μοντέλα! Υπάρχουν πολλά τσιπ που κάνουν αυτή τη δουλειά και το καθένα έχει διαφορετικές προδιαγραφές. Ο ακόλουθος πίνακας συγκεντρώνει τα διασημότερα τσιπάκια αυτής της κατηγορίας:

Συντονιστής	Συχνότητες
Elonics E4000	52 - 2200 MHz (κενό μεταξύ 1100 MHz και 1250 MHz)
Rafael Micro R820T	24 - 1766 MHz
Fitipower FC0013	22 - 1100 MHz
Fitipower FC0012	22 - 948.6 MHz
FCI FC2580	146 - 924 MHz (κενό μεταξύ 308 MHz και 438 MHz)

Όπως καταλαβαίνετε το Elonics E4000 αποτελεί την καλύτερη επιλογή, αφού υποστηρίζει το ευρύτερο φάσμα συχνοτήτων. Το ADB-S λειτουργεί στα 1090Mhz, αλλά στο μέλλον ενδέχεται να ασχοληθούμε ξανά με το Software Defined Radio και τότε μπορεί να μας ενδιαφέρουν άλλες συχνότητες. Επομένως, για να κάνουμε τη δουλειά τώρα αλλά και στο μέλλον, θα ήταν καλό να επιλέξουμε έναν tuner με το τσιπάκι Elonics E4000 — ή έστω με το Rafael Micro R820T. Μια απλή αναζήτηση με τις λέξεις κλειδιά «RTL2832U E4000» θα εμφανίσει και πάλι αρκετά αποτελέσματα. Τέλος, αν ο TV tuner της επιλογής μας δεν έχει υποδοχή για κανονική κεραία τηλεόρασης και δέχεται μόνο τα μικρά βύσματα IEC, καλό θα ήταν να προμηθευτούμε και τον κατάλληλο μετατροπέα...

Ραδιοκύματα στο Raspberry Pi

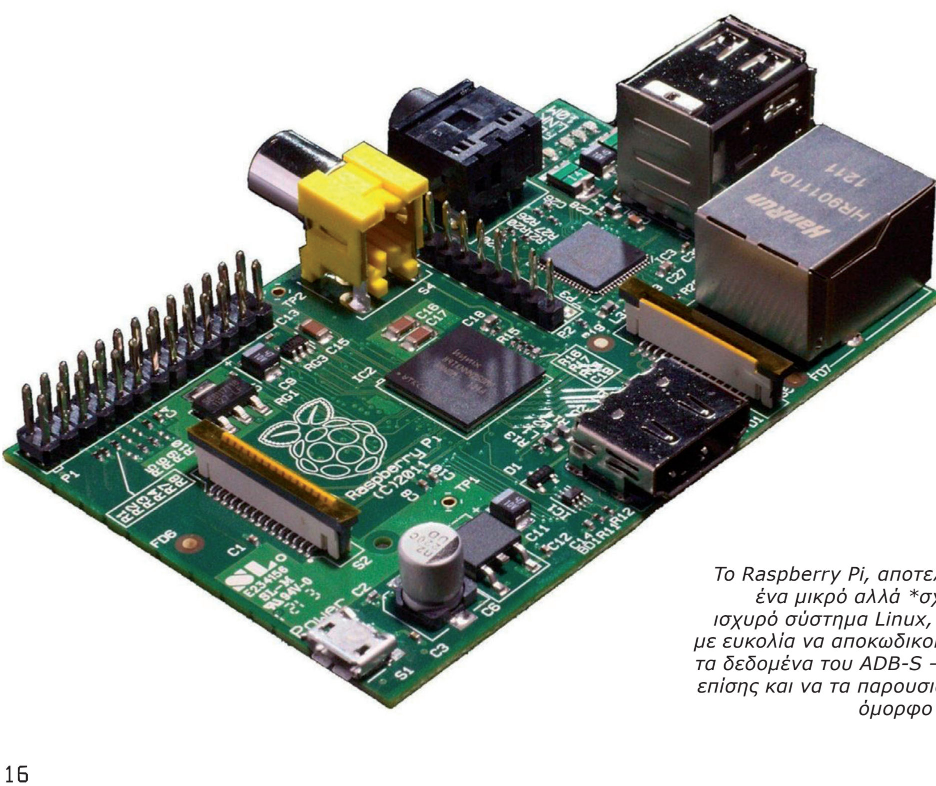
Ξεκινήσαμε εγκαθιστώντας στο Raspberry Pi την τελευταία έκδοση του Raspbian. Φυσικά, αμέσως μετά αναβαθμίσαμε όλα τα πακέτα του συστήματος, δίνοντας πρώτα ένα apt-get update και στη συνέχεια ένα apt-get upgrade. Καλό θα ήταν να κάνετε κι εσείς το ίδιο, στη διανομή που επιλέξατε να χρησιμοποιήσατε. Μετά την προετοιμασία του συστήματος, σειρά έχει η εγκατάσταση των απαιτούμενων προγραμμάτων. Αρχικά, πρέπει να εγκαταστήσουμε τον ειδικό driver για το τσιπάκι RTL του TV tuner. Ό,τι χρειαζόμαστε βρίσκεται στο πακέτο RTL-SDR (`http://sdr.osmocom.org/trac/wiki/rtl-sdr`) και υποκαθιστά πλήρως τον εργοστασιακό driver της συσκευής μας. Για την εγκατάσταση του RTL-SDR δεν προσφέρεται κάποιο DEB package, αλλά

1. Έχουμε «άγριες» διαθέσεις :D

τα πράγματα είναι αρκετά απλά. Δείτε πώς το προσθέσαμε στο Raspbian:

```
sudo apt-get install autoconf libtool libusb-1.0-0-dev
git clone git://git.osmocom.org/rtl-sdr.git
cd rtl-sdr/
autoreconf -i
./configure
make
sudo make install
sudo make install-udev-rules
sudo ldconfig
```

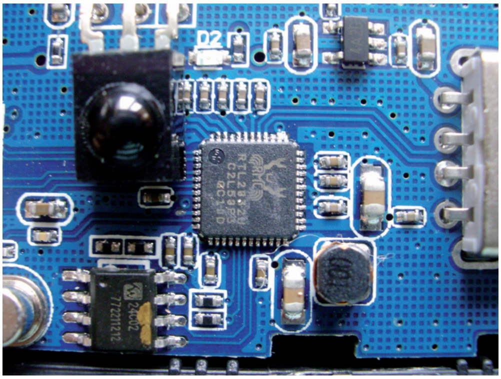
Σαν πρώτο βήμα, εγκαταστήσαμε στο σύστημα μερικά πακέτα που απαιτούνται για τη μεταγλώττιση του RTL-SDR. Στη συνέχεια αντιγράψαμε τον πηγαίο κώδικα του driver από το σχετικό git repository, μεταβήκαμε στον κατάλογο με τον κώδικα και πραγματοποιήσαμε την μεταγλώττιση με την λίγο-πολύ γνώριμη διαδικασία (configure, make και install). Εδώ αξίζει να προσέξουμε αυτό το «make install-udev-rules», το οποίο προσθέτει κάποιους κανόνες στο udev του συστήματος. Έτσι, το πρόγραμμα θα μπορεί να έχει πρόσβαση στο δίαυλο USB και στον TV tuner, χωρίς να το εκτελούμε με δικαιώματα root. Κρίσιμο βήμα αποτελεί και η εκτέλεση του ldconfig, καθώς ενημερώνει το σύστημα για τις νέες βιβλιοθήκες που προστέθηκαν με την εγκατάσταση του RTL-SDR. Αυτή η ενημέρωση αποτελεί αναγκαία προϋπόθεση για το επόμενο στάδιο της εργασίας μας. Βλέπετε, μετά την εγκατάσταση του ειδικού driver για τον TV tuner, σειρά έχει το πρόγραμμα που θα λαμβάνει τα σήματα του ADS-B:



*Το Raspberry Pi, αποτελώντας ένα μικρό αλλά *σχετικά* ισχυρό σύστημα Linux, μπορεί με ευκολία να αποκωδικοποιήσει τα δεδομένα του ADB-S — όπως επίσης και να τα παρουσιάσει με όμορφο τρόπο.*

```
cd ~
git clone https://github.com/antirez/dump1090.git
cd dump1090
make
```

Όπως βλέπετε, ούτε γι’ αυτό το πρόγραμμα υπάρχει κάποιο πακέτο εγκατάστασης. Ωστόσο, η διαδικασία είναι εξαιρετικά απλή. Αφού κατεβάσουμε τον κώδικα από το σχετικό git repository, μπαίνουμε στον αντίστοιχο κατάλογο και πραγματοποιούμε την εγκατάσταση δίνοντας ένα απλό make. Αν περιμένατε κάτι περισσότερο, θα σας απογοητεύσουμε :D Το dump1090 αναλαμβάνει τα πάντα! Επιτυγχάνει το συντονισμό στις συχνότητες του ADS-B, αποκωδικοποιεί τα σχετικά σήματα και φροντίζει να τα παρουσιάσει με όμορφο και κατανοητό τρόπο. Πλέον, είμαστε έτοιμοι να αρχίσουμε την παρακολούθηση της εναέριας κυκλοφορίας!



*Το μικρό αλλά θαυματουργό Realtek RTL2832U. Χάρη σ’ αυτό και με τη βοήθεια κάποιων *πειραγμένων* drivers, μπορούμε να συντονιστούμε σε όποια συχνότητα επιτρέπει το τσιπάκι συντονισμού κι όχι μόνο σ’ αυτές της τηλεόρασης.*

Αρχική δοκιμή

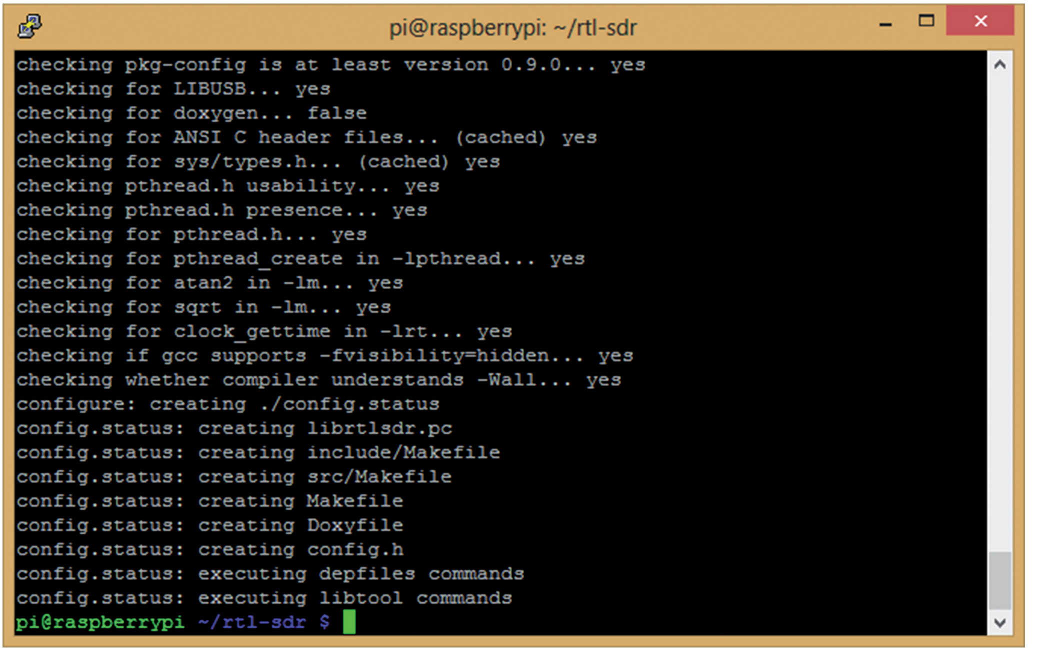
Πριν προχωρήσουμε οφείλουμε να σταθούμε σε μια σημαντική λεπτομέρεια. Το τροφοδοτικό του Raspberry Pi έχει αρκετά μικρή ισχύ. Έτσι, τη στιγμή που συνδέουμε τον TV tuner η τάση τροφοδοσίας γονατίζει, με συνέπεια το σύστημα να πραγματοποιεί ένα βίαιο hardware reset! Για αυτό το λόγο, πρέπει πρώτα να απενεργοποιήσουμε τη συσκευή (δίνοντας ένα sudo poweroff), μετά να συνδέσουμε τον TV tuner και μόνο τότε να ενεργοποιήσουμε το σύστημα ξανά. Προφανώς, όσοι χρησιμοποιείτε κάποιον κανονικό υπολογιστή, μπορείτε να αγνοήσετε ό,τι διαβάσατε μόλις :P

Μπορούμε τώρα να δοκιμάσουμε τη λειτουργία του driver, δίνοντας κάτι τέτοιο:

```
$ rtl_test –t
Found 1 device(s):
  0:  ezcap USB 2.0 DVB-T/DAB/FM dongle

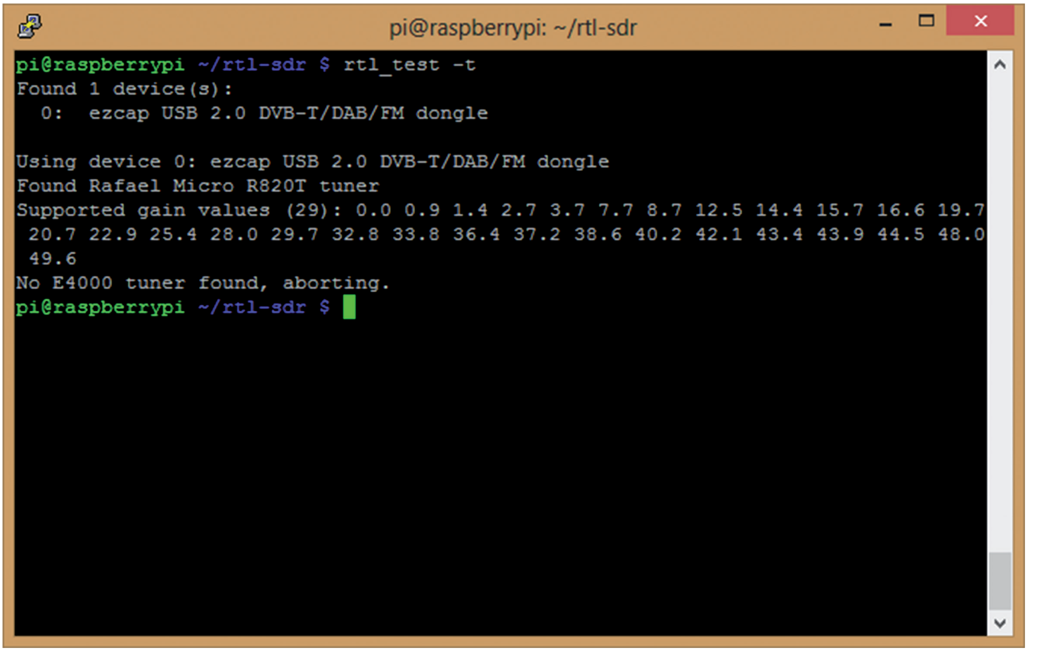
Using device 0: ezcap USB 2.0 DVB-T/DAB/FM dongle
Found Rafael Micro R820T tuner
Supported gain values (29): 0.0 0.9 1.4 2.7 3.7 7.7 8.7 12.5 14.4 15.7
16.6 19.7 20.7 22.9 25.4 28.0 29.7 32.8 33.8 36.4 37.2 38.6 40.2 42.1 43.4
43.9 44.5 48.0 49.6
No E4000 tuner found, aborting.
```

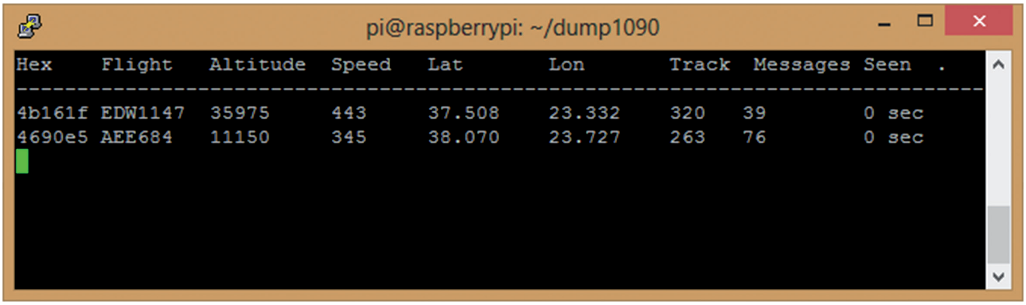
Τα μηνύματα που προκύπτουν από την εκτέλεση του rtl_test ποικίλουν, ανάλογα με τον ακριβή τύπο της συσκευής που έχουμε προμηθευτεί. Παρατηρείστε ότι στη δική μας δοκιμή το rtl_test παραπονέθηκε ότι δεν εντόπισε το τσιπάκι E4000. Αυτό το μήνυμα μπορείτε να το αγνοήσετε! Το πρόγραμμα στηρίζει ορισμένες από τις δοκιμές του στο συγκεκριμένο τσιπ κι αν ο TV tuner χρησιμοποιεί κάποιο άλλο, δεν υπάρχει κανένας λόγος ανησυχίας. Η μόνη περίπτωση που θα πρέπει να ανησυχήσουμε, είναι αν με την εκτέλεση του trl_test δεν εντοπιστεί καμία συσκευή.



Μεταγλωττίζοντας το RTL-SDR... Αυτή η συλλογή προγραμμάτων επιτρέπει στο σύστημά μας να συντονίσει τον TV tuner στις συχνότητες που επιθυμούμε!

Η πρώτη δοκιμή του RTL-SDR ήταν επιτυχής. Το TV tuner εντοπίστηκε με επιτυχία και είμαστε πλέον έτοιμοι ν' αρχίσουμε την παρακολούθηση των αεροσκαφών :D





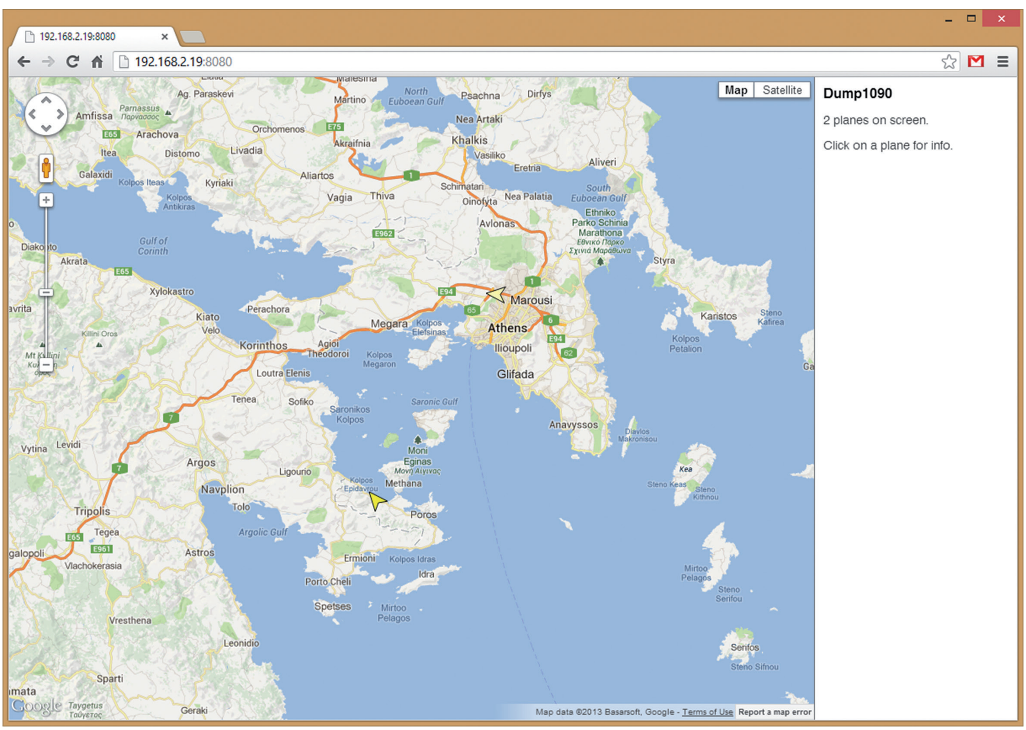
Τα πρώτα αεροπλάνα που εντόπισε ο δέκτης μας, με την εργοστασιακή κεραία. Μπορούμε να δούμε αριθμό πτήσης, ταχύτητα σε κόμβους αλλά και τη θέση (με συντεταγμένες GPS).

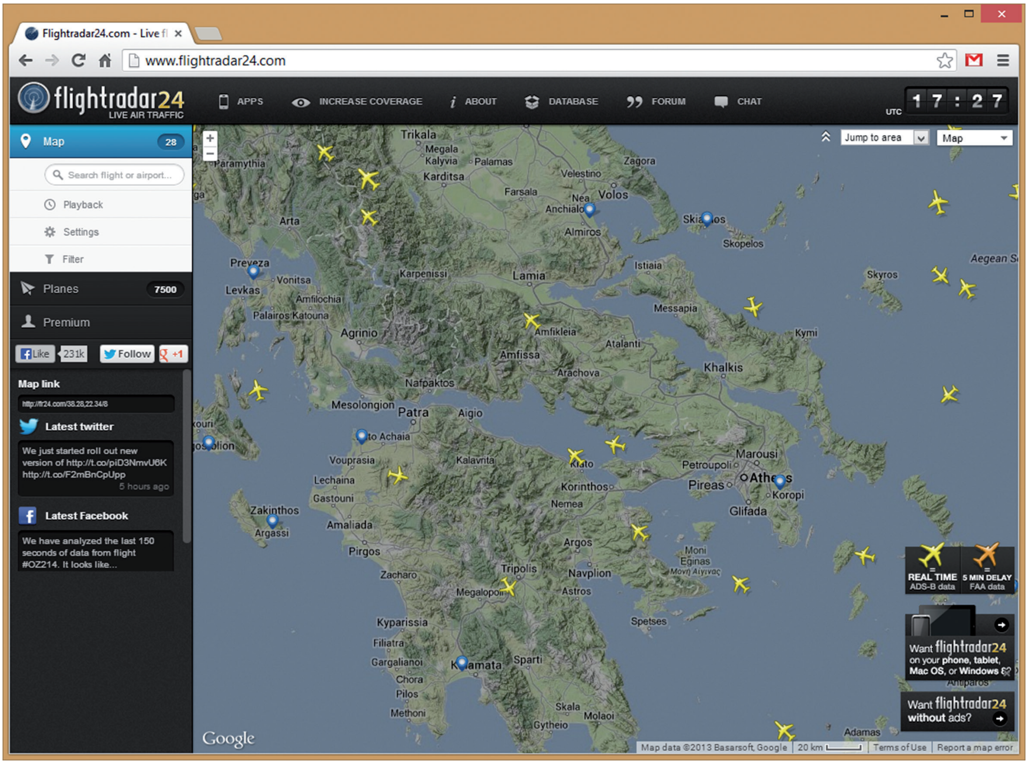
Σήματά σου καθαρά, δυνατά!

Εφόσον όλα λειτουργούν σωστά, μπορούμε να αρχίσουμε τη λήψη των πακέτων ADS-B! Όπως αναφέραμε και πριν, όλη η δουλειά θα γίνει από το dump1090. Το εν λόγω πρόγραμμα μπορεί να παρουσιάσει τα δεδομένα του με δύο τρόπους: Είτε στην κονσόλα, με απλό κείμενο, είτε μέσω ενός τοπικού web server που ξεκινά αυτόματα και σερβίρει τα δεδομένα μέσα από ένα web interface. Ας ξεκινήσουμε με τα απλά:

```
./dump1090 --interactive
```

Ποιος ασχολείται με τερματικά, όταν μπορεί να έχει μια ωραιότατη ιστοσελίδα με τον χάρτη της; Το dump1090 ενσωματώνει έναν μικρό webserver, ο οποίος δείχνει ζωντανά τη θέση των αεροσκαφών που έχουν εντοπιστεί.





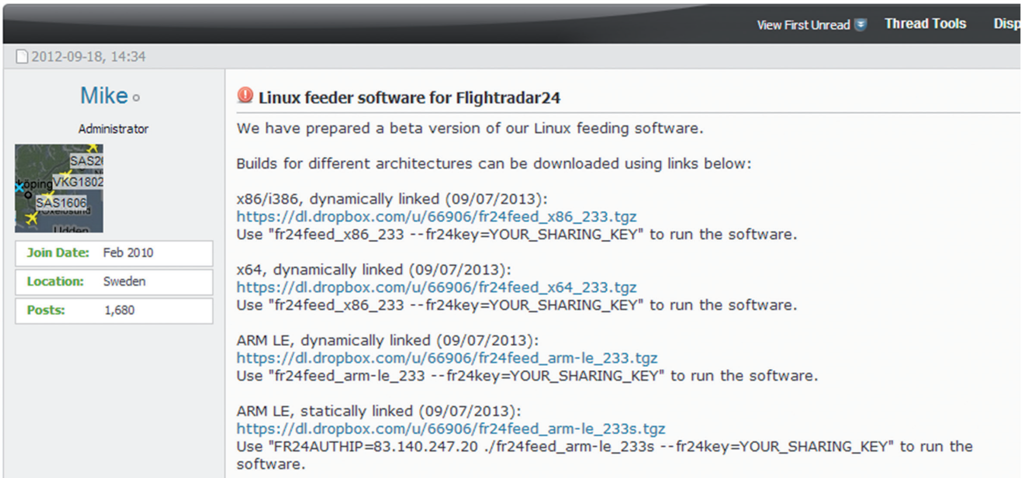
Οι υπηρεσίες που εμφανίζουν ζωντανά τις θέσεις των αεροσκαφών σε χάρτη, χρησιμοποιούν το ADB-S και βασίζονται σε εθελοντές που δωρίζουν τα δεδομένα των δεκτών τους. Μια από αυτές είναι και το flightradar24.

Καλώντας το dump1090 με αυτόν τον τρόπο, τα δεδομένα για τις διερχόμενες πτήσεις θα εμφανιστούν στην κονσόλα. Έτσι θα δούμε διάφορες γραμμές, στις οποίες θα προβάλλονται πληροφορίες όπως ο δεκαεξαδικός κωδικός του εκάστοτε αεροπλάνου, ο αριθμός πτήσης, το ύψος και οι συντεταγμένες από το GPS του σκάφους! Επιπρόσθετα, μπορούμε να δούμε πόσα μηνύματα έχουμε λάβει από κάθε αεροπλάνο, πριν από πόση ώρα λάβαμε το τελευταίο και πάει λέγοντας.

Ωραία η κονσόλα, δε λέμε, αλλά είναι και λίγο βαρετή. Για να πάρουμε τις πληροφορίες με έναν πιο όμορφο τρόπο, αρκεί να ξεκινήσουμε το πρόγραμμα με τον ακόλουθο τρόπο:

```
./dump1090 --net
```

Τώρα μπορούμε να ανοίξουμε τον αγαπημένο μας browser και να επισκεφθούμε τη διεύθυνση `http://raspberrypi:8080`. Προφανώς, στη θέση του raspberrypi θα πρέπει να βάλετε τη διεύθυνση IP του συστήματος με το Linux ή, αν πρόκειται για το ίδιο σύστημα, τη λέξη localhost. Η σελίδα που θα εμφανιστεί περιλαμβάνει έναν χάρτη από το Google Maps, επάνω στον οποίο εμφανίζονται όλα τα αεροπλάνα που έχει εντοπίσει το πρόγραμμα! Η θέση κάθε αεροπλάνου ανανεώνεται κάθε λίγα δευτερόλεπτα. Στα δεξιά της σελίδας βλέπουμε πόσα αεροπλάνα έχουν εντοπιστεί, καθώς και τις σχετικές πληροφορίες για το καθένα. Τέλος, αν θέλουμε τόσο τη γραφική


View First Unread | Thread Tools | Disp

2012-09-18, 14:34

Mike
Administrator

Join Date: Feb 2010
Location: Sweden
Posts: 1,680

Linux feeder software for Flightradar24

We have prepared a beta version of our Linux feeding software.

Builds for different architectures can be downloaded using links below:

x86/i386, dynamically linked (09/07/2013):
https://dl.dropbox.com/u/66906/fr24feed_x86_233.tgz
Use "fr24feed_x86_233 --fr24key=YOUR_SHARING_KEY" to run the software.

x64, dynamically linked (09/07/2013):
https://dl.dropbox.com/u/66906/fr24feed_x64_233.tgz
Use "fr24feed_x86_233 --fr24key=YOUR_SHARING_KEY" to run the software.

ARM LE, dynamically linked (09/07/2013):
https://dl.dropbox.com/u/66906/fr24feed_arm-le_233.tgz
Use "fr24feed_arm-le_233 --fr24key=YOUR_SHARING_KEY" to run the software.

ARM LE, statically linked (09/07/2013):
https://dl.dropbox.com/u/66906/fr24feed_arm-le_233s.tgz
Use "FR24AUTHIP=83.140.247.20 ./fr24feed_arm-le_233s --fr24key=YOUR_SHARING_KEY" to run the software.

Οι εκδόσεις του flightradar24 είναι τέσσερις και καλύπτουν τις διασημότερες αρχιτεκτονικές. Εμείς δεν ασχοληθήκαμε με τους x86/x64. Πήραμε την έκδοση για ARM και συγκεκριμένα εκείνη που τρέχει στο Raspberry Pi.

αναπαράσταση των δεδομένων, όσο κι εκείνη με τη μορφή κειμένου, μπορούμε να συνδυάσουμε τις δύο παραμέτρους που δέχεται το dump1090 ως εξής:

```
./dump1090 --interactive --net
```

Δωρεές... πακέτων

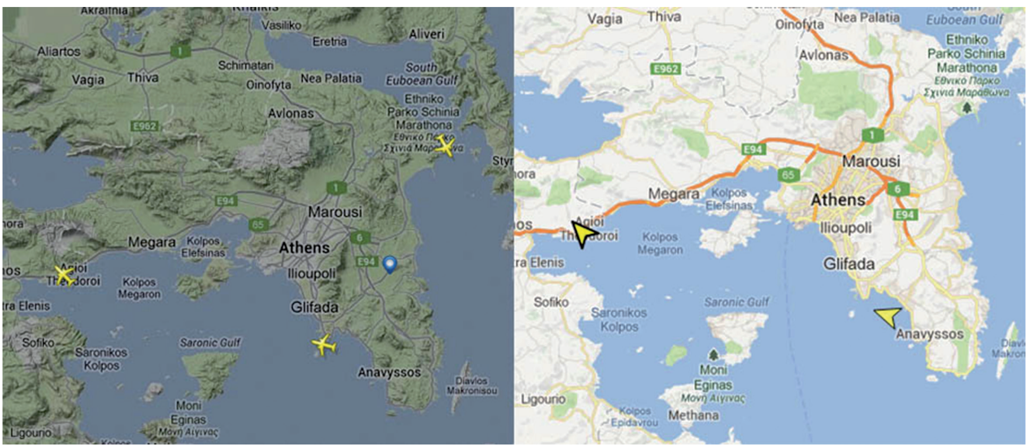
Πλέον μπορούμε να βλέπουμε τα αεροπλάνα που πετάνε πάνω απ' το κεφάλι μας, χωρίς καν να σηκωθούμε από την καρέκλα του γραφείου μας! Αρκετά ενδιαφέρον, αλλά θα είχε περισσότερη πλάκα αν μοιραζόμασταν αυτά τα δεδομένα με τον υπόλοιπο κόσμο. Υπάρχουν αρκετές υπηρεσίες που συγκεντρώνουν τέτοιου είδους δεδομένα και μία από αυτές είναι το Flightradar24 (www.flightradar24.com). Η συγκεκριμένη υπηρεσία παρέχει τον δικό της client για την αποστολή δεδομένων (ονομάζεται και feeder). Μπορούμε να βρούμε τις δοκιμαστικές εκδόσεις αυτού του προγράμματος για το Linux, στην ακόλουθη θέση: http://forum.flightradar24.com/threads/4270-Linux-feeder-software-for-Flightradar24. Η κατάλληλη έκδοση για το Raspberry Pi είναι η «ARM LE, dynamically linked» και παρέχεται μεταγλωττισμένη, έτοιμη προς χρήση.

```
cd ~
mkdir fr24feed
cd fr24feed
wget https://dl.dropbox.com/u/66906/fr24feed_arm-le_233.tgz
tar –zxvf fr24feed_arm-le_233.tgz
rm fr24feed_arm-le_233.tgz
```

Τώρα μένει να ξεκινήσουμε το dump1090 και να το συνδέσουμε *κάπως* με τον client (feeder) για την υπηρεσία Flightradar24. Για το σκοπό αυτό, αρκεί να δώσουμε τα ακόλουθα:

```
cd ~
./dump1090/dump1090 --net --net-sbs-port 30003 –interactive
./fr24feed/fr24feed_arm-le_225 --fr24key=FR24_KEY --bs-ip=127.0.0.1 --bs-port=30003
```

Με τα παραπάνω, το dump1090 θα ξεκινήσει τον τοπικό του webserver κι επιπρόσθετα θα περιμένει συνδέσεις στη δικτυακή θύρα 30003. Το fr24feed θα συνδεθεί στη συγκεκριμένη θύρα και θα αρχίσει να αντλεί δεδομένα από τον server. Σημειώστε ότι στη θέση του FR24_KEY θα πρέπει να βάλουμε το κλειδί που μας παρείχαν οι άνθρωποι την υπηρεσίας. Για να προμηθευτούμε ένα τέτοιο κλειδί αρκεί να στείλουμε email στο support@fr24.com. Στο email θα πρέπει να δώσουμε τον κωδικό ICAO του πλησιέστερου αεροδρομίου, τον οποίο μπορούμε να βρούμε κοιτώντας το output της εντολής dump1090 —net.



Συγκρίνοντας τους δύο χάρτες, μπορούμε να βεβαιωθούμε ότι ο δέκτης μας δουλεύει όπως θα έπρεπε!

Δημιουργία κεραίας

Με την κεραία που συνόδευε τον TV tuner, καταφέραμε να πιάσουμε το σήμα από πολλά αεροπλάνα. Ωστόσο, κατασκευάζοντας τη δική μας κεραία θα έχουμε σίγουρα πολύ καλύτερα αποτελέσματα. Βλέπετε, η δική μας κεραία θα είναι απόλυτα προσαρμοσμένη στη συχνότητα του ADB-S και θα παρουσιάζει υψηλότερη απόδοση. Η κεραία που χρειαζόμαστε είναι μια «διπολική, μισού μήκους κύματος» και για την κατασκευή της θα χρειαστούμε ένα μικρό κομμάτι ξύλου ή φελλού, λίγο σκληρό σύρμα, ένα ομοαξονικό καλώδιο κεραίας[2] και το σχετικό βύσμα για τη σύνδεση στον TV tuner.

Σε γενικές γραμμές, η κατασκευή έχεις ως εξής: Το κομμάτι του ξύλου ή του φελλού θα αποτελέσει τη βάση της κεραίας, επάνω στην οποία θα «καρφώσουμε» δύο κομμάτια σύρματος. Τα δύο σύρματα πρέπει να στηριχτούν σε μικρή απόσταση μεταξύ τους και τα τμήματα που εξέχουν θα πρέπει να τα λυγίσουμε σε ορθή γωνία, κατά τέτοιο τρόπο ώστε να «δείχνουν» σε αντίθετες κατευθύνσεις. Εδώ χρειάζεται λίγη προσοχή! Τα δύο σύρματα θα πρέπει να έχουν ακριβώς το ίδιο μήκος. Επιπρόσθετα, η απόσταση από το άκρο του ενός ως το άκρο του άλλου θα πρέπει να ισούται με 13,8

2. Μπορούμε να το ζητήσουμε σαν «καλώδιο κεραίας 75 Ohm»!

εκατοστά. Αυτό το μέγεθος αποτελεί το μισό μήκος κύματος που αντιστοιχεί στη συχνότητα του ADB-S (1090MHz)[3]. Έτσι, αν έχουμε καρφώσει τα δύο σύρματα σε απόσταση 3,8 εκατοστών, το λυγισμένο τμήμα του καθενός θα πρέπει να έχει μήκος 5 εκατοστά (2*5+3,8=13,8). Προφανώς, αν η απόσταση των σημείων στήριξης των συρμάτων είναι μεγαλύτερη, τα λυγισμένα τμήματα θα πρέπει να είναι λίγο μικρότερα και το αντίστροφο. Ωστόσο, καλό θα ήταν να στερεώσουμε τα δύο σύρματα σε όσο το δυνατόν μικρότερη απόσταση. Στα άκρα των συρμάτων που έχουμε στερεώσει επάνω στη βάση της κεραίας θα πρέπει να συνδέσουμε και τους δύο αγωγούς του ομοαξονικού καλωδίου. Τέλος, στην άλλη άκρη του καλωδίου θα πρέπει να συνδέσουμε το βύσμα και στη συνέχεια να το προσαρτήσουμε στον TV tuner.

Εμείς κατασκευάσαμε μια κεραία ακολουθώντας τις παραπάνω οδηγίες και πρέπει να σας πούμε ότι πετύχαμε εντυπωσιακά αποτελέσματα! Στις δοκιμές μας καταφέραμε να εντοπίσουμε αεροπλάνα που βρίσκονταν έως και 80km μακριά από τη βάση μας. Αρκεί να σας πούμε ότι το Raspberry Pi με τον TV tuner και τη κεραία μας βρίσκονταν κάπου στα βόρεια της Αθήνας, ενώ ορισμένα αεροπλάνα που εντοπίσαμε βρίσκονταν έξω απ’ την Μαλεσίνα!



Η μικρή μας κεραία απαρτίζεται από έναν φελλό κρασιού (ροζέ), λίγο σύρμα από καλώδια ρεύματος οικοδομής και (το ένα άκρο) από ένα παλιό καλώδιο κεραίας.

Επίλογος

Ένας TV tuner με το RTL2832U κι ο driver RTLSDR ανοίγουν ένα νέο παράθυρο για τα χόμπι μας: Μας εισάγουν στον κόσμο του Software-Defined Radio! Εκτός από τα πακέτα ADS-B είναι δυνατό να λάβουμε και πακέτα AIS, που είναι τα αντίστοιχα για τις θέσεις των πλοίων. Μπορούμε επίσης να λάβουμε δεδομένα για pagers (ή βομβητές, όπως λέγονται στην Ελλάδα), καθώς επίσης και να ακούσουμε συνομιλίες από ασυρμάτους[4]! Με λίγο κόπο θα μπορούσαμε να πιάσουμε και τα σήματα του International Space Station, του Voyager αλλά και του Mars Rover :D Κάτι τέτοιο βέβαια απαιτεί πιο περίπλοκες εγκαταστάσεις, όπως τα κατάλληλα δορυφορικά πιάτα. Ωστόσο, η λήψη τέτοιων σημάτων αποτελεί ένα πρώτης τάξεως παράδειγμα για τις απεριόριστες δυνατότητες που προσφέρει το Software-Defined Radio. Για την ώρα, πάντως, θα αρκεστούμε στα (χαρούμενα) αεροπλάνα.

3. Το μήκος κύματος για αυτή τη συχνότητα είναι 27,52cm. Έτσι, το μισό μήκος κύματος ισούται με 13,76cm, που είναι κατά προσέγγιση ίσο με 13,8cm.
4. Από CB ή και άλλες, άλλων συχνοτήτων… Πάντως *γκουχ* όχι *γκουχ* της αστυνομίας! *γκουχ*

Skill: Advanced
Tags: cloud storage, sync, remote control, Dropbox, Python



by multiPetros

Απομακρυσμένος έλεγχος μέσω cloud storage

Πολλές από τις σύγχρονες ιντερνετικές υπηρεσίες είναι μάλλον εξειδικευμένες και η χρήση τους δεν παρεκκλίνει πολύ από τον σκοπό για τον οποίο αρχικά δημιουργήθηκαν. Σχεδόν πάντα έτσι είναι τα πράγματα — αλλά όσοι μας διαβάζετε καιρό ήταν σαν ν' ακούσατε την έμφαση στη λέξη «σχεδόν». Σε λίγο, φίλες και φίλοι, θα διαπιστώσουμε πως κάποιες υπηρεσίες cloud storage μπορούν να χρησιμοποιηθούν ακόμη και για τον απομακρυσμένο έλεγχο του υπολογιστή μας!

Όλοι οι πάροχοι υπηρεσιών online storage που σέβονται τον εαυτό τους, προσφέρουν στους χρήστες τη δυνατότητα αυτόματου συγχρονισμού. Στην πλειονότητα των περιπτώσεων, αυτό επιτυγχάνεται με την εγκατάσταση και τη ρύθμιση ενός κατάλληλου client. Αυτό το πρόγραμμα παρακολουθεί τους τοπικούς αλλά και τους απομακρυσμένους φακέλους και φροντίζει ώστε κάθε στιγμή να έχουμε τις πιο πρόσφατες εκδόσεις των αρχείων μας, τόσο στον υπολογιστή μας όσο και στα μηχανήματα της υπηρεσίας. Άραγε, θα μπορούσε κανείς να χρησιμοποιήσει μια υπηρεσία online storage, καθώς και το αντίστοιχο πρόγραμμα συγχρονισμού, για να πετύχει κάτι εντελώς διαφορετικό από που διαφημίζει/πουλάει η εκάστοτε εταιρεία; Προφανώς η ερώτησή μας είναι ρητορική, αφού αν η απάντηση ήταν αρνητική αυτό το άρθρο δεν θα είχε γραφτεί ποτέ ;) Για να μη μακρηγορούμε, σκοπεύουμε να σας αποδείξουμε ότι με τη βοήθεια μιας τέτοιας υπηρεσίας θα πετύχουμε την απομακρυσμένη διαχείριση ενός υπολογιστή! Όπως και κάθε άλλη φορά, δεν θα περιοριστούμε στην απλή περιγραφή της λογικής. Πολύ περισσότερο, θα γράψουμε και τον απαιτούμενο κώδικα που θα μας βοηθήσει να πετύχουμε το σκοπό μας...

Η λογική του εγχειρήματος

Χρησιμοποιώντας τη βασική λειτουργία που προσφέρουν οι online storage clients, δηλαδή το συγχρονισμό αρχείων, μπορούμε να μεταφέρουμε εντολές ανάμεσα σε δύο ή περισσότερα συστήματα. Θα εφαρμόσουμε τη λογική client-server, όπου ο ένας υπολογιστής θα λειτουργεί ως αποστολέας τον εντολών (client) κι ο άλλος ως αποδέκτης (server). Ως μέσο μεταφοράς των εντολών θα χρησιμοποιήσουμε απλά αρχεία κειμένου! Ο κώδικας, από την πλευρά του client, θα γράφει τις εντολές σε ειδικά αρχεία. Από τη μεριά του server, το πρόγραμμα θα ανοίγει τα ειδικά αρχεία και θα εκτελεί τις εντολές που βρίσκει στο εσωτερικό τους. Με παρόμοιο τρόπο θα στέλνει τις απαντήσεις του ο server, σχετικά με την εξέλιξη των εργασιών που εκτελεί. Ένας από τους παράγοντες που πρέπει να λάβουμε υπόψη και ο οποίος θα επηρεάσει τη λογική της εφαρμογής μας, είναι το γεγονός ότι ο συγχρονισμός των αρχείων δεν πραγματοποιείται ακαριαία. Έτσι, η επικοινωνία του server με τον client δεν μπορεί να εξελίσσεται σε πραγματικό χρόνο. Τα δύο μέρη θα μιλούν σε προγραμματισμένα τακτά χρονικά διαστήματα και, ουσιαστικά, μόνο τότε θα τσεκάρουν αν υπάρχουν νέες εντολές ή νέα μηνύματα. Απλό σαν σκέψη, δεν συμφωνείτε;

Λίγα λόγια για τον κώδικα

Μια που έχουμε όλα τα απαραίτητα υλικά (δύο υπολογιστές, μια υπηρεσία on-line storage και τον client που αναλαμβάνει το συγχρονισμό), μπορούμε να προχωρήσουμε άμεσα στην κατασκευή των προγραμμάτων μας. Για την υλοποίηση θα καταφύγουμε στη γλώσσα προγραμματισμού Python. Εκτός του ότι πρόκειται για μια διαπλατφορμική γλώσσα —χαρακτηριστικό που την καθιστά ιδανική για το συγκεκριμένο project—, η αλήθεια είναι ότι μάς έχει γίνει ιδιαιτέρως αγαπητή από τα σχετικά άρθρα του Sonic, εδώ, στο deltaHacker.

Όπως εξηγήσαμε νωρίτερα, θα χρειαστεί να γράψουμε δύο ξεχωριστά προγράμματα: Το ένα θα τρέχει στον server (στον υπολογιστή που θα δέχεται τις εντολές) και το άλλο στον client (τον υπολογιστή από τον οποίο θα στέλνουμε τις εντολές). Τα δύο προγράμματα θα τρέχουν στην κονσόλα και η μεταξύ τους επικοινωνία θα εξελίσσεται σαν ένας (πολιτισμένος) διάλογος. Αρχικά, ο χρήστης θα εισάγει τις εντολές του, στη συνέχεια ο client θα τις στέλνει στον server κι όταν λίγο αργότερα λαμβάνει κάποια απάντηση, θα την εμφανίζει στο χρήστη!

Οι εντολές που θα υποστηρίζει η εφαρμογή μας είναι οι απολύτως βασικές για το χειρισμό αρχείων (λίστα αρχείων, αντιγραφή, μετονομασία, μετακίνηση και διαγραφή, δημιουργία και διαγραφή καταλόγων), δύο ακόμα για το κατέβασμα αρχείων και μια για τη λήψη ενός στιγμιότυπου της οθόνης, αν στο απομακρυσμένο σύστημα υπάρχει περιβάλλον γραφικών). Φυσικά το ρεπερτόριο των εντολών θα μπορούσε να επεκταθεί απεριόριστα, σύμφωνα με τις ανάγκες και τις επιθυμίες του καθενός. Εμείς θα δημιουργήσουμε μια βασική εκδοχή των δύο προγραμμάτων και σας προ(σ) καλούμε να τα επεκτείνετε με όποιον τρόπο θέλετε :) Με τον μηχανισμό που περιγράψαμε θα μπορούσαμε να εκτελούμε στο απομακρυσμένο σύστημα ακόμη και shell commands. Ωστόσο, κάτι τέτοιο θα συνιστούσε μεγάλο και ιδιαίτερα επικίνδυνο κενό ασφάλειας.

Για την ανάγκες του άρθρου στηριχτήκαμε στη δημοφιλή υπηρεσία Dropbox και φυσικά στο αντίστοιχο πρόγραμμα συγχρονισμού. Ωστόσο, τα προγράμματα που θα παρουσιάσουμε μπορούν να λειτουργήσουν με οποιαδήποτε παρόμοια υπηρεσία.

Αποδέκτης εντολών

Θα ξεκινήσουμε από το πρόγραμμα που θα εκτελείται στο απομακρυσμένο σύστημα και το οποίο θα λαμβάνει και θα εκτελεί τις εντολές μας. Ο κώδικας ξεκινά με τη δήλωση των απαραίτητων βιβλιοθηκών:

```
import time
import os
import shutil
import urllib
import ImageGrab
```

Η βιβλιοθήκη time περιέχει τη συνάρτηση sleep, με την οποία θα δημιουργούμε τις απαραίτητες χρονικές καθυστερήσεις, μεταξύ των διαδοχικών ελέγχων για τη λήψη εντολών. Οι βιβλιοθήκες os και shutil περιλαμβάνουν τις συναρτήσεις remove, rmdir, rename, makedirs, listdir, copy, isdir και isfile, που απαιτούνται για την υλοποίηση των «εντολών» που υποστηρίζει ο μηχανισμός μας. Η βιβλιοθήκη urllib περιέχει τη συνάρτηση urlretrieve κι απαιτείται για την υλοποίηση της εντολής που θα κατεβάζει αρχεία. Τέλος, η βιβλιοθήκη ImageGrab ενσωματώνει τη συνάρτηση grab, με τη βοήθεια της οποία θα λαμβάνουμε στιγμιότυπα της οθόνης. Σημειώστε ότι για την εισαγωγή αυτής της βιβλιοθήκης πρέπει να έχουμε εγκατεστημένο το πακέτο PIL της Python (Python Image Library).

Ο κώδικας συνεχίζει με τη δήλωση τριών μεταβλητών, τις οποίες πρέπει να τροποποιήσουμε ανάλογα με το σετάρισμα του δικού μας μηχανήματος:

```
commands_file = "C:\\Users\\myname\\Dropbox\\commands.droprc"
results_file = "C:\\Users\\myname\\Dropbox\\results.droprc"
delay = 30
```

Οι δύο πρώτες περιέχουν τον κατάλογο που συγχρονίζεται μέσω Dropbox, καθώς εκεί θα αποθηκεύεται το αρχείο με τις προς εκτέλεση εντολές (commands.droprc) αλλά κι εκείνο με τα αποτελέσματα της εκτέλεσής τους (results.droprc). Η μεταβλητή delay καθορίζει το χρονικό διάστημα μεταξύ των ελέγχων για νέες εντολές. Εμείς επιλέξαμε τα 30 δευτερόλεπτα, αλλά αν η δραστηριότητά σας είναι περιορισμένη ή

δεν σας ενδιαφέρει η άμεση εκτέλεση των εντολών, μπορείτε να αυξήσετε το διάστημα. Φυσικά, μπορείτε και να το μειώσετε, αλλά προσέξτε να μην το παρακάνετε και «μπουκώσετε» το σύστημα! Ας δούμε τώρα τις συναρτήσεις που υλοποιούν τις εντολές, τις οποίες θα υποστηρίζει ο μηχανισμός μας:

```
def cmd_screenshot(path):
    output = []
    img = ImageGrab.grab()
    try:
        img.save(path)
        output.append("Done!")
    except Exception as err:
        output.append(err)
    return output
def cmd_download(url, path):
    output = []
    try:
        urllib.urlretrieve(url, path)
        output.append("Done!")
    except Exception as err:
        output.append(err)
    return output
def cmd_rm(path):
    output = []
    try:
        os.remove(path)
        output.append("Done!")
    except Exception as err:
        output.append(err)
    return output

def cmd_rmdir(path):
    output = []
    try:
        os.rmdir(path)
        output.append("Done!")
    except Exception as err:
        output.append(err)
    return output
```

```
def cmd_rename(old, new):
    output = []
    try:
        os.rename(old, new)
        output.append("Done!")
    except Exception as err:
        output.append(err)
    return output
def cmd_mkdir(path):
    output = []
    try:
        os.makedirs(path)
        output.append("Done!")
    except Exception as err:
        output.append(err)
    return output

def cmd_ls(path):
    output = []
    if not(os.path.isdir(path)):
        output.append("Directory " + path + " not exist!")
        return output
    output.append("Directory " + path + " contents:")
    if not(path.endswith("/")) or not(path.endswith("\\")):
        path = path + "/"
    elements = os.listdir(path)
    for element in elements:
        if os.path.isfile(path + element):
            output.append(element + " - " + str(os.path.getsize(path + 
element)) + " bytes")
        else:
            output.append(element + " - directory")
    return output
def cmd_copy(source, dest):
    output = []
    if not(os.path.isfile(source)):
        output.append("Source file not found")
```

```
            return append
        try:
            shutil.copy(source, dest)
            output.append("Done!")
        except Exception as err:
            output.append(err)
        return output
```

Όπως θα παρατηρήσατε, όλες οι συναρτήσεις των εντολών επιστρέφουν μια λίστα, με τα αποτελέσματα της εκτέλεσης της εκάστοτε εντολής. Ο κώδικας συνεχίζει με τη συνάρτηση main()[1] καθώς και τον έλεγχο της ειδικής μεταβλητής __name__. Η συνάρτηση main αποτελεί τον πυρήνα του προγράμματός μας και περιέχει έναν ατέρμονα βρόχο, ο οποίος ελέγχει για την ύπαρξη νέων εντολών, φροντίζει για την εκτέλεσή τους και δημιουργεί το αρχείο με τα σχετικά αποτελέσματα:

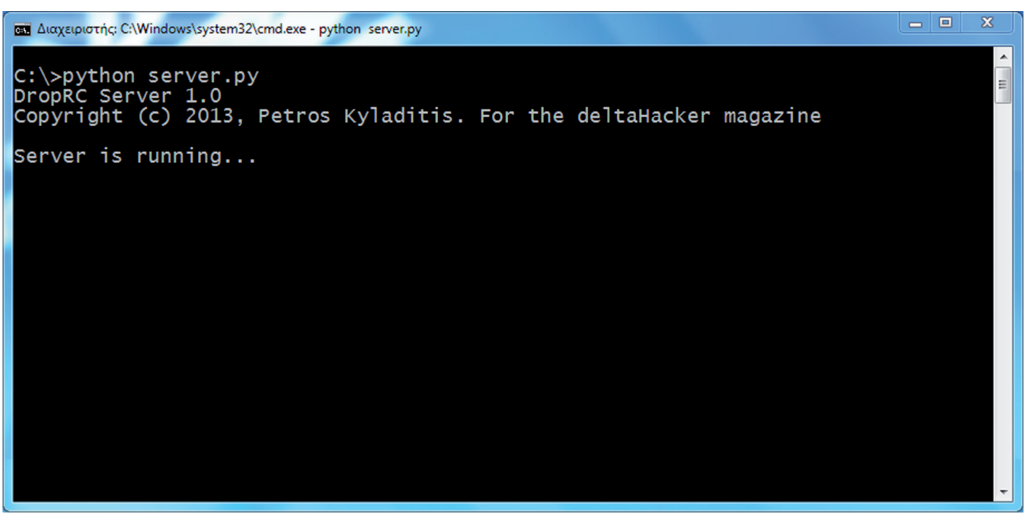
```
    def main():
        print "DropRC Server 1.0 - Copyright (c) 2013, Petros Kyladitis. For the deltaHacker magazine\n"
        while True:
            if os.path.exists(commands_file):
                output = ""
                commands = []
                fcmd = open(commands_file, 'r')
                lines = fcmd.read()
                commands = lines.split("\n")
                fcmd.close()
                for command in commands:
                    if command.startswith("screenshot"):
                        cmd = command.split(" ", 1)
                        output = output + command + "\n"
                        output = output + '\n'.join(cmd_screenshot(cmd[1]))
                        output = output + "\n\n"
                    elif command.startswith("download"):
                        cmd = command.split(" ", 2)
                        output = output + command + "\n"
                        output = output + '\n'.join(cmd_download(cmd[1], cmd[2]))
                        output = output + "\n\n"
```

1. Εάν δε γνωρίζετε τι είναι η συνάρτηση main(), αυτό είναι μια καλή ένδειξη για το ότι οφείλετε να διαβάσετε τα άρθρα του Sonic περί Python, στα τεύχη 005 έως και 011 του περιοδικού. Κι αν δεν έχετε κάποια εξ αυτών, εκμεταλλευτείτε τις καταπληκτικές τιμές που ισχύουν για την ηλεκτρονική έκδοση του περιοδικού: http://deltahacker.gr/pdf12order

```
    elif command.startswith("rm"):
        cmd = command.split(" ", 1)
        output = output + command + "\n"
        output = output + '\n'.join(cmd_rm(cmd[1]))
        output = output + "\n\n"
    elif command.startswith("rmdir"):
        cmd = command.split(" ", 1)
        output = output + command + "\n"
        output = output + '\n'.join(cmd_rmdir(cmd[1]))
        output = output + "\n\n"
    elif command.startswith("rename"):
        cmd = command.split(" ", 2)
        output = output + command + "\n"
        output = output + '\n'.join(cmd_rename(cmd[1], cmd[2]))
        output = output + "\n\n"
    elif command.startswith("mkdir"):
        cmd = command.split(" ", 1)
        output = output + command + "\n"
        output = output + '\n'.join(cmd_mkdir(cmd[1]))
        output = output + "\n\n"
    elif command.startswith("ls"):
        cmd = command.split(" ", 1)
        output = output + command + "\n"
        output = output + '\n'.join(cmd_ls(cmd[1]))
        output = output + "\n\n"
    elif command.startswith("copy"):
        cmd = command.split(" ", 2)
        output = output + command + "\n"
        output = output + '\n'.join(cmd_copy(cmd[1], cmd[2]))
        output = output + "\n\n"
    elif command == "":
        pass
    else:
        output = output + command + "\n"
        output = output + "Command not found!\n\n"
foutcontents = ""
```

```
            for outline in output:
                foutcontents = foutcontents + outline
            fout = open(results_file, 'a')
            fout.write(foutcontents)
            fout.close()
            os.remove(commands_file)
        time.sleep(delay)
if __name__ == "__main__":
    main()
```

Αρχικά γίνεται ένας έλεγχος για την ύπαρξη αρχείου με εντολές. Αν βρεθεί τέτοιο αρχείο, το πρόγραμμα αρχίζει να το διαβάζει γραμμή προς γραμμή, να μεταφράζει και να εκτελεί τις εντολές. Η συνάρτηση startswith ελέγχει αν η εκάστοτε γραμμή του αρχείου ξεκινάει με κάποια από τις υποστηριζόμενες εντολές και προχωρά στην εκτέλεση της αντίστοιχης ρουτίνας. Χρησιμοποιούμε τη συνάρτηση startswith διότι μια εντολή μπορεί να συνοδεύεται από μια ή περισσότερες παραμέτρους και αυτό συνεπάγεται ότι οι γραμμές με τις εντολές δεν θα έχουν πάντα την ίδια ακριβώς μορφή. Ο διαχωρισμός της εντολής από τις παραμέτρους πραγματοποιείται με τη συνάρτηση split. Με αυτή τη συνάρτηση χωρίζουμε την εκάστοτε γραμμή σε επιμέρους συμβολοσειρές (strings), σύμφωνα με κάποιον χαρακτήρα διαχωρισμού (στην περίπτωσή μας είναι το κενό). Αφού ολοκληρωθεί η εκτέλεση όλων των εντολών που βρέθηκαν στο σχετικό αρχείο, το πρόγραμμα δημιουργεί ένα νέο, με τα αποτελέσματα. Ταυτόχρονα, το αρχείο με τις εντολές διαγράφεται! Στη συνέχεια, με τη βοήθεια της συνάρτησης time.delay εισάγουμε μια χρονική καθυστέρηση, έως ότου πραγματοποιηθεί ο νέος έλεγχος για την ύπαρξη αρχείου με εντολές...



Το πρόγραμμα στον απομακρυσμένο υπολογιστή εκτελείται και περιμένει καρτερικά και χωρίς πολλά — πολλά για τις εντολές του χρήστη, από τον άλλο υπολογιστή. Οι εντολές καταφθάνουν με τη βοήθεια του προγράμματος συγχρονισμού της υπηρεσίας cloud storage (Dropbox ;))

Προγραμματίζοντας από την μεριά του client

Έχοντας ολοκληρώσει τον κώδικα για το απομακρυσμένο μηχάνημα, μπορούμε να ξεκινήσουμε το πρόγραμμα για τον υπολογιστή από τον οποίο θα στέλνουμε τις εντολές. Ο κώδικας γι' αυτό το πρόγραμμα ξεκινά και πάλι με τη δήλωση των απαραίτητων βιβλιοθηκών:

```
import time
import os
import re
import datetime
```

Η χρησιμότητα των δύο πρώτων είναι γνωστή από τον κώδικα του άλλου προγράμματος. Η βιβλιοθήκη re περιλαμβάνει συναρτήσεις κανονικών εκφράσεων[2], τις οποίες θα χρησιμοποιήσουμε για την εύκολη επεξεργασία των εντολών που εισάγει ο χρήστης (διαχωρισμός εντολών και παραμέτρων σε επιμέρους τμήματα). Από τη βιβλιοθήκη datetime θα χρησιμοποιήσουμε τις συναρτήσεις για τη λήψη της ώρας και της ημερομηνίας, ώστε ο έλεγχος για τυχόν αποτελέσματα να πραγματοποιείται στο σωστό χρόνο. Το πρόγραμμα συνεχίζεται με τη δήλωση ορισμένων βασικών μεταβλητών:

```
commands_file = "C:\\Users\\myname\\Dropbox\\commands.droprc"
results_file = "C:\\Users\\myname\\Dropbox\\results.droprc"
delay = 30
valid_commands = ["screenshot", "download", "rm", "rmdir", "rename", "mkdir",
"ls", "copy"]
valid_params = [1, 2, 1, 1, 2, 1, 1, 2]
```

Οι τρεις πρώτες μεταβλητές είναι και πάλι γνωστές, από το άλλο πρόγραμμα. Μια παρατήρηση που πρέπει να κάνουμε είναι το γεγονός ότι τόσο στον server, όσο και στον client, οι μεταβλητές που περιέχουν τα αρχεία εντολών κι αποτελεσμάτων πρέπει να δείχνουν στην ίδια θέση, εντός του συγχρονιζόμενου φακέλου. Η μεταβλητή valid_commands αποτελεί μια λίστα με τις εντολές που υποστηρίζει ο server. Αντίστοιχα, η valid_params περιλαμβάνει μια λίστα με το πλήθος των απαιτούμενων παραμέτρων για κάθε εντολή.

Ακολουθεί η συνάρτηση addSecs, η οποία προσθέτει στην τρέχουσα ώρα τόσα δευτερόλεπτα, όσα έχουμε ορίσει με την παράμετρο secs και επιστρέφει ένα αντικείμενο datetime. Από το αντικείμενο αυτό εξάγουμε την ώρα σε ευανάγνωστη μορφή, ώστε να παρουσιάσουμε στον χρήστη την ώρα του επόμενου ελέγχου για τη λήψη αποτελεσμάτων...

```
def addSecs(secs):
    now = datetime.datetime.now().time()
    checkdate = datetime.datetime(100, 1, 1, now.hour, now.minute,
now.second)
    checkdate = checkdate + datetime.timedelta(seconds=secs)
```

2. Περισσότερα για τις κανονικές εκφράσεις μπορείτε να μάθετε από τα σχετικά άρθρα που φιλοξενούμε στα τεύχη 021, 022 και σ' αυτό που τώρα διαβάζετε.

```
    return checkdate.time()
```

Το πρόγραμμα ολοκληρώνεται με τη συνάρτηση main() και τον έλεγχο της __name__:

```
def main():
    print "DropRC Client 1.0 - Copyright (c) 2013, Petros Kyladitis. For the deltaHacker magazine\n"
    while True:
        if not(os.path.exists(commands_file)) and not(os.path.exists(results_file)):
            user_input = ""
            user_cmd = ""
            print "Listening commands. At new line, enter '.' to end the input, or '?' to see acceptable commands."
            while user_cmd != ".":
                user_cmd = raw_input("> ")
                splitted_cmd = re.split(''' (?=(?:[^"]|'[^']*'|"[^"]*")*$)''', user_cmd)
                if splitted_cmd[0] in valid_commands:
                    if valid_params[valid_commands.index(splitted_cmd[0])] == (len(splitted_cmd) - 1):
                        user_input = user_input + user_cmd + "\n"
                    else:
                        print "Invalid number of parameters! Expected: " + str(valid_params[valid_commands.index(splitted_cmd[0])])
                elif user_cmd == "?":
                    for i in range(0, len(valid_commands)):
                        print valid_commands[i] + " -> expecting " + str(valid_params[i]) + " param(s)."
                else:
                    print "Command not found! Valid commands: " + ", ".join(valid_commands)
            try:
                print "Writing commands file"
                fcommands = open(commands_file, 'w')
                fcommands.write(user_input)
                fcommands.close()
                print "Done!\n"
            except Exception as err:
                print err
```

```
        else:
            if os.path.exists(results_file):
                fresults = open(results_file, 'r')
                lines = fresults.read()
                fresults.close()
                print lines
                os.remove(results_file)
            print "Waiting for server response (next check at " +
str(addSecs(delay)) + ")..."
            time.sleep(delay)
if __name__ == "__main__":
    main()
```

Η μέθοδος main αποτελείται κι εδώ από έναν μεγάλο βρόγχο. Αρχικά, γίνεται έλεγχος για την ύπαρξη αρχείου με εντολές. Αν δεν υπάρχει αρχείο εντολών, το πρόγραμμα προτρέπει το χρήστη να εισαγάγει τις επιθυμητές. Για κάθε εντολή που εισάγει, το πρόγραμμα ελέγχει αν αυτή υποστηρίζεται από την εφαρμογή καθώς και το αν περιλαμβάνει το προβλεπόμενο πλήθος παραμέτρων. Στην περίπτωση που η είσοδος του χρήστη δε συμβαδίζει με τις απαιτήσεις της εφαρμογής, προβάλλεται ένα σχετικό μήνυμα-ειδοποίηση. Ο χρήστης μπορεί να ενημερωθεί για τις υποστηριζόμενες εντολές δίνοντας σε μια γραμμή το χαρακτήρα του λατινικού ερωτηματικού «?». Το τέλος των εντολών σηματοδοτείται με την εισαγωγή σε μια νέα γραμμή του χαρακτήρα της τελείας «.».

Διαχειριστής: C:\Windows\system32\cmd.exe - python client.py

```
DropRC Client 1.0
Copyright (c) 2013, Petros Kyladitis. For the deltaHacker magazine

Listening commands. At new line, enter '.' to end the input, or '?' to see accep
table commands.
> download https://bitbucket.org/multipetros/office13tweaks/downloads/Office13Tw
eaks-2.0_bin.zip c:/office.zip
> ?
screenshot -> expecting 1 param(s).
download -> expecting 2 param(s).
rm -> expecting 1 param(s).
rmdir -> expecting 1 param(s).
rename -> expecting 2 param(s).
mkdir -> expecting 1 param(s).
ls -> expecting 1 param(s).
copy -> expecting 2 param(s).
> ls c:\
> .
Command not found! Valid commands: screenshot, download, rm, rmdir, rename, mkdi
r, ls, copy
Writing commands file
Done!

Waiting for server response (next check at 04:03:15)...
```

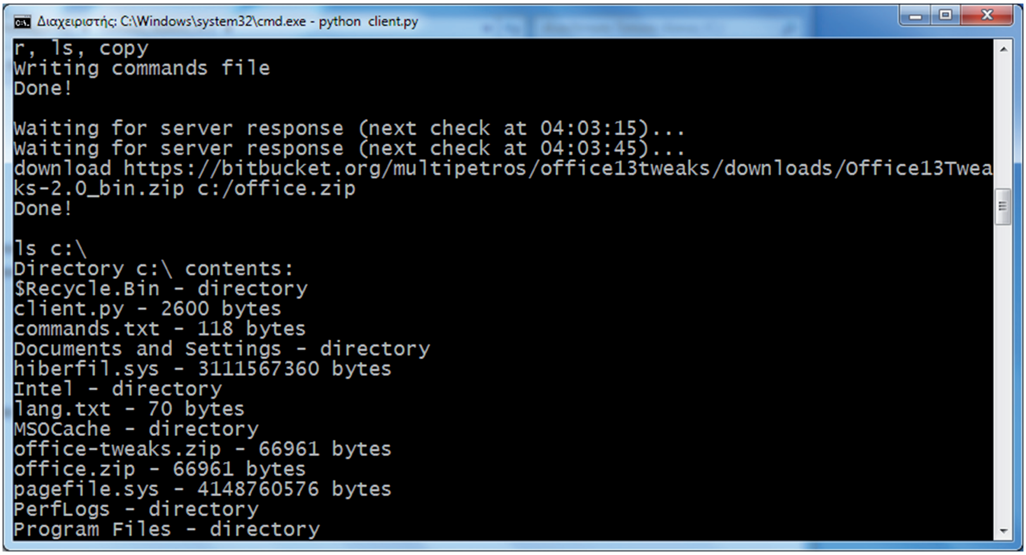
Ο χρήστης δίνει τις εντολές του: Το κατέβασμα ενός προγράμματος tweaking για το Office2013, καθώς κι ένα listing του c:\. Στο ενδιάμεσο, συμβουλεύεται και την ενσωματωμένη «βοήθεια» του προγράμματος, για να θυμηθεί τις υποστηριζόμενες εντολές και τις παραμέτρους που δέχεται. Όταν το πρόγραμμα εντοπίζει αρχείο με αποτελέσματα, σπεύδει να τα προβάλλει στο χρήστη. Διαφορετικά, εισάγεται μια χρονική καθυστέρηση και το πρόγραμμα ενημερώνει τον χρήστη για το πότε θα γίνει ο επόμενος έλεγχος.

Το ταξίδι ή ο προορισμός;

Όπως είδατε, η κατασκευή μιας εφαρμογής που επιτρέπει τον έλεγχο ενός υπολογιστή μέσω υπηρεσιών on-line storage, δεν είναι ιδιαίτερα δύσκολη. Φυσικά, το τελικό αποτέλεσμα δεν μπορεί να συγκριθεί με τις δυνατότητες που προσφέρει μια εφαρμογή remote desktop. Το ίδιο ισχύει κι όταν αρχίσουμε να σκεφτόμαστε θέματα ασφαλείας κι αναλογιζόμαστε λύσεις του στιλ «απομακρυσμένη πρόσβαση μέσω SSH». (Γίνεται, ακόμη κι όταν το μηχάνημα στο οποίο συνδεόμαστε τρέχει Windows — βλ. http://deltahacker.gr/windowsopenssh). Όμως ο σκοπός του άρθρου δεν ήταν αυτός. Αυτό που επιδιώξαμε, πέρα από τη διασκέδαση με τη συγγραφή του κώδικα, ήταν να δείξουμε τι μπορεί να πετύχει κανείς όταν σκέφτεται έξω από τα τείχη! Για τη δική σας ευκολία, μπορείτε να κατεβάσετε τα αρχεία με τον κώδικα («server» για το μηχάνημα που δέχεται τις εντολές και «client» για το μηχάνημα στο οποίο βρισκόμαστε) από τη διεύθυνση

http://bit.ly/dh023droprc

Σας προτείνουμε να μελετήσετε τον κώδικα, να πειραματιστείτε, να τον επεκτείνετε και φυσικά να μοιραστείτε μαζί μας τις ιδέες σας, τα προβλήματα που συναντήσατε αλλά και τις λύσεις. Κι όσο για το ερώτημα που θέσαμε στην αρχή της παραγράφου, θα αφήσουμε τον Καβάφη να απαντήσει για εμάς: «…κι αν πτωχική τη βρεις, η Ιθάκη δεν σε γέλασε, έτσι σοφός που έγινες, με τόση πείρα, ήδη θα το κατάλαβες οι Ιθάκες τι σημαίνουν…»!



Η απάντηση του server στην εκτέλεση των εντολών του client, καταφθάνει μετά από λίγο. Όπως βλέπουμε, όλες τους εκτελέστηκαν πιστά! Πλέον, ο client μπορεί να προχωρήσει στην αποστολή νέων εντολών...

Skill: Intermediate
Tags: encryption, cryptography, Enigma, Python



by Sonic2000gr

The Enigma Reborn - Μέρος 3ο: Πολύς κόπος για το τίποτα!

Η Enigma φαίνεται ότι δεν έχει πλέον μυστικά από εμάς. Στο άρθρο του τεύχους 022 ολοκληρώσαμε τη λειτουργία του rotor assembly. Έχουμε στα χέρια μας μια μηχανή με τρεις ρότορες και τη δυνατότητα να επιλέξουμε τον τύπο και τη θέση του καθενός. Η μηχανή μας, επίσης, κρυπτογραφεί —απόλυτα πιστά προς την πρωτότυπη— κάθε κείμενο που της δίνουμε. Έχουμε καταφέρει μάλιστα να προγραμματίσουμε και τα bugs, εχμ, δηλαδή τα features της αρχικής έκδοσης, όπως το περίφημο double stepping.

Μη βιαστείτε να πείτε όμως ότι έχουμε ολοκληρώσει την εξομοίωση μας, γιατί μας μένουν ακόμα αρκετά πράγματα...

- Τα περίφημα ring settings, μια ρύθμιση της μηχανής που προκαλεί αρκετή σύγχυση όπως θα δείτε, γιατί πολλά άρθρα στο web τα μπερδεύουν με τις αρχικές θέσεις (rotor starting positions) που έχουμε δει ήδη.
- Το plugboard, αυτό το περίφημο σύστημα με τα ζεύγη καλωδίων που τοποθετούνται στο εμπρός μέρος της μηχανής κι ανακατεύουν την είσοδο των γραμμάτων (αλλά και την έξοδο στο lampboard, όπως θα δούμε).
- Ένα περιβάλλον γραφικών για να μπορούμε να δείξουμε τη μηχανή στους φίλους μας χωρίς να δεχτούμε πάλι πειράγματα του τύπου «όλη μέρα μπροστά σε μια μαύρη οθόνη με αριθμούς που τρέχουν, είσαι» (Εσείς φταίτε που δεν μαθαίνετε στους φίλους σας τη μαγεία του terminal, όμως.)

Είναι γεγονός ότι η μηχανή που έχουμε μέχρι στιγμής δεν έχει δα και πολύ σοβαρή κρυπτογράφηση: Τρεις ρότορες από 26 συνδυασμούς ο καθένας μας δίνουν 26x25x26 = 16900 δυνατότητες — και, όχι, το 25 δεν είναι τυπογραφικό λάθος: είναι το double stepping! Όπως την έχουμε δει ως τώρα, η Enigma πωλούνταν σε επιχειρήσεις για κρυπτογράφηση σημαντικής αλληλογραφίας. Ναι, ακόμα και στις δεκαετίες του 30 και 40, η βιομηχανική κατασκοπεία ήταν γεγονός. Από αυτή την εμπορικής χρήσης μηχανή απουσίαζε εντελώς το plugboard, ενώ στις στρατιωτικές Enigma υπήρχε πάντα και χρησιμοποιούταν υποχρεωτικά.

Και λέμε υποχρεωτικά, γιατί η συνδεσμολογία του plugboard είναι τέτοια ώστε αν δεν χρησιμοποιήσουμε κανένα καλώδιο τότε όλα τα γράμματα αντιστοιχίζονται στον εαυτό τους κι έχουμε μια απλή Enigma εμπορικής χρήσης, όπως αυτή που έχουμε φτιάξει μέχρι στιγμής. Οι Γερμανοί όμως δεν ήταν τόσο χαζοί: είχαν ήδη κάνει μια πρώτη κρυπτανάλυση της «κομμένης» αυτής Enigma και ήξεραν ότι παρά την πολυπλοκότητα των κινούμενων μερών της, η κρυπτογράφηση μόνο με τρεις ρότορες ήταν ασθενική. Σε καμιά περίπτωση δεν θα ήθελαν να καταφέρει ο Alan Turing να τη σπάσει, πίνοντας το πρωινό του καφεδάκι. (Είμαστε σίγουροι ότι πέρασε αρκετά ευχάριστα απογεύματα για να το καταφέρει. Καλό αυτό, μιας και δεν είχε FreeBSD ο άνθρωπος!)

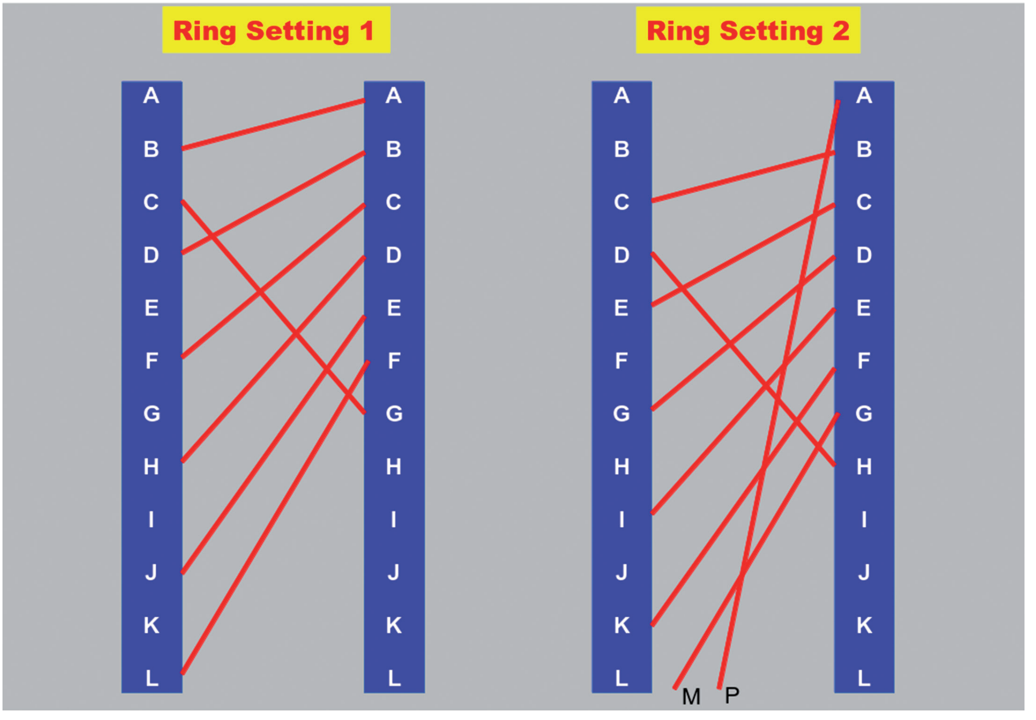
Ring Settings — Τι μπέρδεμα είν' αυτό;

Μπορεί να διαβάσετε αρκετά άρθρα και να δείτε κάμποσα βίντεο στο YouTube, όπου κάποιος παρουσιάζοντας μια Enigma αλλάζει θέση στους ρότορες και μιλάει για ring settings. Όχι: Τα ring settings δεν έχουν να κάνουν με τις αρχικές θέσεις (τις οποίες βάζουμε περιστρέφοντας τους ρότορες μέσα στη μηχανή από τα thumbwheels, πριν αρχίσουμε την κρυπτογράφηση ή την αποκρυπτογράφηση ενός μηνύματος).

Μα, τι είναι τότε τα ring settings και τι χρησιμότητα έχουν; Ας πάρουμε ένα παλιό καλό μας φίλο: τον ρότορα τύπου III. Δεν περιμένω να θυμόσαστε τη συνδεσμολογία του, αλλά είναι η ακόλουθη:

Type III Rotor (Ring Setting 1)

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
B	D	F	H	J	L	C	P	R	T	X	V	Z	N	Y	E	I	W	G	A	K	M	U	S	Q	O



Πώς επηρεάζουν τα ring settings την κρυπτογράφηση; Στο σχήμα μας φαίνεται τμήμα ενός ρότορα τύπου III, αρχικά με ring setting 1 (το default, θα λέγαμε). Περιστρέφοντάς τον στο ring setting 2, ουσιαστικά περιστρέφουμε μόνο την εσωτερική καλωδίωση κατά μια θέση. Έτσι, το καλώδιο που ένωνε το A→B ενώνει τώρα το B→C. Στη θέση A, το καλώδιο που ένωνε το Z→O, τώρα ενώνει το A→P. Είναι προφανές ότι αυτή η απλή μετατόπιση δεν αποτελεί πραγματική, νέα κρυπτογράφηση (όπως αυτή που θα επιτυγχάναμε αν, για παράδειγμα, αλλάζαμε το ρότορα με έναν διαφορετικού τύπου).

Φανταστείτε τώρα ότι τα καλώδια που κάνουν όλες αυτές τις συνδέσεις μπορούν να περιστραφούν ανεξάρτητα από την αριστερή και τη δεξιά πλάκα του ρότορα και να μετατοπιστούν όσες θέσεις θέλουμε εμείς. Σαν να έχει ο ρότορας ένα εσωτερικό δακτυλίδι, που να περιστρέφει μόνο την καλωδίωση. Τι θα γινόταν αν το κάνατε αυτό; Ας υποθέσουμε ότι γυρίζετε αυτό το δαχτυλίδι κατά μια θέση:

- Το καλώδιο που ένωνε το A→B θα βρεθεί τώρα να ενώνει το B→C. Το καλώδιο που ένωνε το B→D θα ενώνει τώρα το C→E και πάει λέγοντας. Προφανώς, το τελευταίο καλώδιο, αυτό που ένωνε το Z→O, θα ενώνει τώρα το A→P! Για να οπτικοποιήσετε καλύτερα το παραπάνω, δείτε απλά το αντίστοιχο σχήμα.
- Καθώς περιστρέφεται μόνο η εσωτερική καλωδίωση, η ρύθμιση Ring Setting δεν έχει καμιά επίδραση στην εγκοπή του ρότορα: η ενεργοποίηση του επόμενου ρότορα εξακολουθεί να γίνεται στο ίδιο ακριβώς σημείο. Ένας ρότορας τύπου III ενεργοποιεί τον επόμενο (αριστερότερό του) ρότορα κατά την περιστροφή του από V→W.

Και γιατί να μπουν οι Γερμανοί σε όλη αυτή τη φασαρία, να φτιάξουν τα Ring Settings; Δυστυχώς (ή μάλλον ευτυχώς!) φίλοι μου, όλο αυτό βασίστηκε σε μια πλάνη. Πίστεψαν ότι έτσι ενίσχυαν την κρυπτογράφηση, ενώ στην πραγματικότητα δεν αλλάζει

τίποτα. Από τη στιγμή που το μόνο πράγμα που γίνεται είναι μια απλή μετατόπιση, στην πραγματικότητα η κρυπτογράφηση είναι ίδια: Ένας ρότορας τύπου III στη θέση B με ring setting 1, είναι ίδιος με ένα τύπου III στη θέση A με ring setting 2. Δεν κερδίζουμε κάτι.

Όμως οι Γερμανοί είχαν ένα πρόβλημα να λύσουν — και πίστεψαν ότι τα ring settings μπορούσαν να βοηθήσουν: Ήξεραν ότι το σημαντικό κομμάτι της κρυπτογράφησης βασιζόταν στην άρχικη θέση κάθε ρότορα, καθώς και στα καλώδια του plugboard. Αυτά είναι το πραγματικό «μυστικό κλειδί» της μηχανής και ήθελαν να έχουν τη δυνατότητα να το αλλάζουν καθημερινά. Όμως έπρεπε να μπορούν να στείλουν αυτή την πληροφορία στα υποβρύχια, μέσω ασυρμάτου, χωρίς να διαρρεύσει. Μεγάλο πρόβλημα! Πώς να στείλεις κρυπτογραφημένο, ένα μυστικό κλειδί; Είναι λίγο σαν το πρόβλημα με το αυγό και την κότα: Χρειάζεται να κρυπτογραφήσεις το μυστικό κλειδί που θα στείλεις στον άλλο ώστε να αποκρυπτογράφησει αυτός μετά το πραγματικό μήνυμα που θα του στείλεις. Βλέπετε γιατί επικράτησε η ασυμμετρική κρυπτογραφία στις μέρες μας;

Η ιδέα τους ήταν η ακόλουθη: Θα έδιναν στα υποβρύχια από ένα χαρτί με μια σειρά από ring settings, rotors και plugboard settings για κάθε μέρα του μήνα. Στην αρχή της μέρας, ο χειριστής του Enigma θα έβαζε τα ring settings που αντιστοιχούσαν στην ημερομηνία και θα λάμβανε ένα μήνυμα, το οποίο θα περιείχε τους ρότορες και τις αρχικές τους θέσεις καθώς και τις θέσεις των καλωδίων στο plugboard: το μυστικό κλειδί για το κανονικό μήνυμα. Μετά, θα έβαζε τις νέες αυτές ρυθμίσεις στη μηχανή για να λάβει τα κρυπτογραφημένα μηνύματα της ημέρας.

Ένα τέτοιο χαρτί μπορείτε να δείτε και στη σχετική εικόνα. Αλλά όπως είπαμε ήδη, αυτή η προσπάθεια κρυπτογράφησης του μυστικού κλειδιού δεν ήταν πολύ πετυχημένη: Ο Alan Turing κατάλαβε αμέσως ότι τα ring settings δεν είχαν καμιά σημασία και συνέχισε να πίνει το καφεδάκι και να σπάει τους κωδικούς του Enigma. Κάθε μέρα!

Για να αλλάξει κάποιος τα ring settings σε κάθε ρότορα, θα έπρεπε να τον αφαιρέσει από τη μηχανή και να περιστρέψει την καλωδίωση χρησιμοποιώντας το μικρό έλασμα που θα δείτε στην εικόνα και το... νύχι του!

Όσο και αν δεν προσέφεραν στην κρυπτογράφηση, είναι σημαντικό να εξομοιώσουμε τα ring settings στη μηχανή μας. Θα χρησιμοποιήσουμε Python dictionaries.

Python dictionaries — A (tiny) crash course!

Άν είστε από τους παλιούς αναγνώστες μας θα μας έχετε ακολουθήσει σε διάφορες περιπέτειες με τις δομές δεδομένων της Python: Σίγουρα έχουμε χρησιμοποιήσει λίστες πολλές φορές και tuples κάποιες άλλες. Δεν είχαμε χρησιμοποιήσει όμως μέχρι σήμερα τα περίφημα λεξικά ή dictionaries της Python. Ήρθε η ώρα να το αλλάξουμε αυτό, γιατί θα μας λύσει τα χέρια!

Τι είναι τα dictionaries; Όπως ίσως υπονοεί και το όνομα «λεξικά», είναι μια δομή δεδομένων που χρησιμοποιεί ζεύγη του τύπου «κλειδί : τιμή». Έχοντας ορίσει ένα λεξικό, μπορούμε να ζητήσουμε την τιμή που αντιστοιχεί σε κάποιο κλειδί. Τα κλειδιά σε ένα λεξικό είναι μοναδικά: Δεν μπορεί να εμφανίζεται το ίδιο κλειδί δύο φορές. Μπορεί όμως σε δύο διαφορετικά κλειδιά να αντιστοιχεί η ίδια τιμή. Αυτό σημαίνει ότι, γενικά, τα λεξικά δεν αναστρέφονται — **εκτός** αν και οι τιμές που έχουμε είναι μοναδικές.

Θα τα καταλάβετε όλα αυτά με ένα απλό παράδειγμα. Μπορείτε αν θέλετε να το γράψετε στο interactive mode της Python. Ας ορίσουμε ένα λεξικό:

```
thedict = { 'A' : 'E', 'B' : 'D', 'C' : 'F' }
```

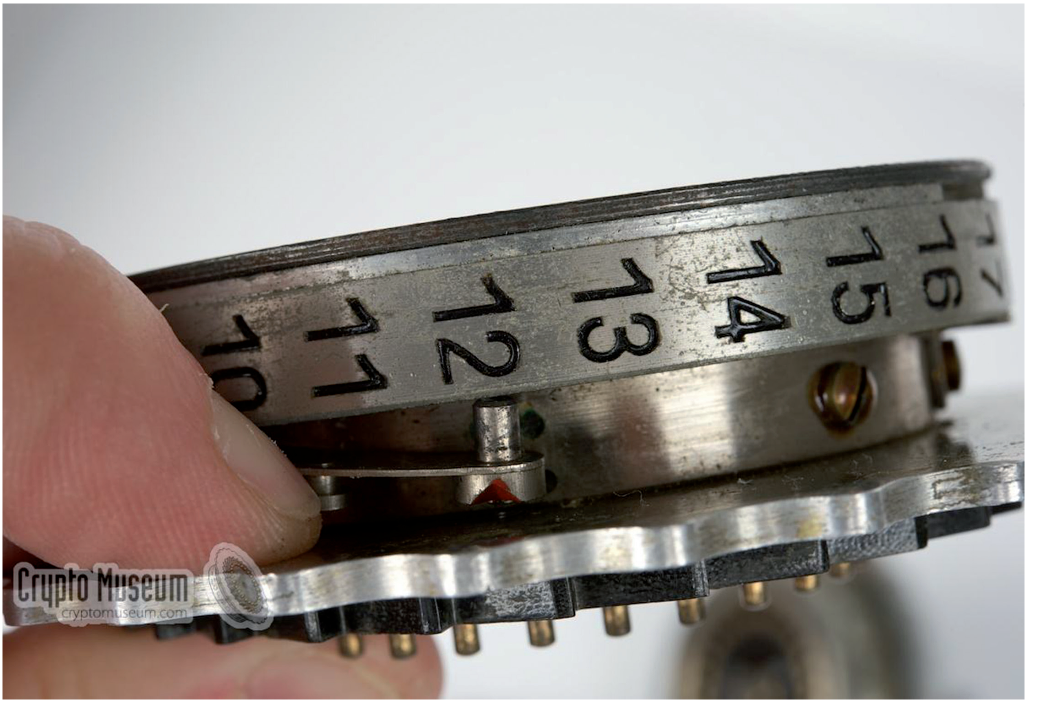
Για να βρούμε τώρα την τιμή που αντιστοιχεί σε ένα κλειδί:

```
print thedict['B']
D
```

Αν ζητήσουμε την τιμή για ένα κλειδί που δεν υπάρχει, θα πάρουμε μήνυμα λάθους:

```
print thedict['G']
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
KeyError: 'G
```

Μπορούμε εύκολα να δούμε αν ένα κλειδί περιέχεται στο λεξικό μας, αποφεύγοντας το προηγούμενο πρόβλημα:



Να πώς αλλάζουν τα ring settings! Το έλασμα που βλέπετε περιστρέφει μόνο το δαχτυλίδι με την καλωδίωση του ρότορα. Η αριστερή και δεξιά πλευρά (με τα γράμματα και την εγκοπή) παραμένουν σταθερές. Οι αριθμοί βοηθούν να ρυθμίσουμε το ring setting στην επιθυμητή τιμή. Μην μπερδεύεστε με τις αρχικές θέσεις (rotor starting positions) που ρυθμίζονται με το ρότορα, πάνω στη μηχανή: Τα ring settings ρυθμίζονται αναγκαστικά με τον ρότορα εκτός μηχανής.

```
print 'A' in thedict
True
print 'G' in thedict
False
```

Η Python παρέχει ένα σωρό ενδιαφέρουσες συναρτήσεις για να δουλέψουμε με λεξικά. Μπορείτε να διαβάσετε τη σχετική τεκμηρίωση:

http://docs.python.org/2/library/stdtypes.html#mapping-types-dict

Περιστρέφοντας τα... δαχτυλίδια

Πρίν αρχίσουμε να παιδευόμαστε με τα dictionaries θα πρέπει να σκεφτούμε πώς θα «περιστρέψουμε» την κρυπτογράφηση ενός ρότορα. Ας υποθέσουμε ότι σκοπεύουμε να βάλουμε σε έναν ρότορα τύπου III το ring setting 2, δηλαδή μετατόπιση μιας θέσης (το ring setting 1 αντιστοιχεί στην κανονική κρυπτογράφηση που έχουμε δώσει ήδη, μέσω της λίστας self.connections).

Έχουμε ήδη μια λίστα self.letter_ring, με όλα τα γράμματα αλφαβητικά:

```
['A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I', 'J', 'K', 'L', 'M', 'N', 'O', 'P', 'Q', 'R', 'S', 'T',
'U', 'V', 'W', 'X', 'Y', 'Z' ]
```

Έχουμε και την αντίστοιχη λίστα connections:

```
['B', 'D', 'F', 'H', 'J', 'L', 'C', 'P', 'R', 'T', 'X', 'V', 'Z', 'N', 'Y', 'E', 'I', 'W', 'G', 'A',
'K', 'M', 'U', 'S', 'Q', 'O']
```

Η ολίσθηση της κρυπτογράφησης είναι να πάρουμε τη δεύτερη αυτή λίστα και να μετατοπίσουμε τις τιμές τις κατά ένα, φτιάχνοντας μια νέα προσαρμοσμένη λίστα (adjusted_connections):

```
['C', 'E', 'G', 'I', 'K', 'M', 'D', 'Q', 'S', 'U', 'Y', 'W', 'A', 'O', 'Z', 'F', 'J', 'X', 'H', 'B',
'L', 'N', 'V', 'T', 'R', 'P']
```

Μόνο που αυτή η δεύτερη λίστα αντιστοιχεί σε ένα self.letter_ring, που ξεκινάει πλέον από το B (adjusted_letter):

```
['B', 'C', 'D', 'E', 'F', 'G', 'H', 'I', 'J', 'K', 'L', 'M', 'N', 'O', 'P', 'Q', 'R', 'S', 'T', 'U',
'V', 'W', 'X', 'Y', 'Z', 'A']
```

Η συνάρτηση μας, με το εμπνευσμένο όνομα rotateRings, είναι η ακόλουθη και εκτελεί και τις δύο αυτές λειτουργίες:

```
def rotateRings(self):
  self.adjusted_letter = self.adjusted_letter[self.ring_setting-1:]+
                             self.adjusted_letter[0:self.ring_setting-1]
  self.adjusted_connections = []
  for i in self.connections:
    neword = ord(i)+ self.ring_setting - 1
    if neword > 90:
      neword = neword - 26
    self.adjusted_connections.append(chr(neword))
```

Προφανώς, θα χρειαστεί να δίνουμε το ring_setting στην init του ρότορα (και θα δώσουμε ως default την τιμή 1, που δεν επιφέρει καμιά αλλαγή).

Μπορούμε να φτιάξουμε τώρα ένα λεξικό που να συνδυάζει αυτές τις δύο λίστες, την adjusted_letter ως κλειδιά και την adjusted_connections ως τιμές. Το λεξικό αυτό θα μπορούμε να το χρησιμοποιήσουμε άμεσα στη συνάρτηση cipher:

B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A
C	E	G	I	K	M	D	Q	S	U	Y	W	A	O	Z	F	J	X	H	B	L	N	V	T	R	P

Έχουμε δύο λίστες, πώς θα τις συνδυάσουμε σε ένα λεξικό; Εδώ έρχεται η Python με τη δυνατότητα dictionary comprehension (αντίστοιχο του list comprehension, όπου δημουργούμε μια λίστα με συγκεκριμένες τιμές σε μια γραμμή κώδικα) και την επίσης χρήσιμη συνάρτηση zip.

Επειδή τρεις γραμμές κώδικα αξίζουν όσο μια παράγραφος περιγραφής, δείτε τι κάνει η zip όπως τη δείχνει το παράδειγμα της επίσημης τεκμηρίωσης:

```
x = [1, 2, 3]
y = [4, 5, 6]
print zip(x, y)
[(1, 4), (2, 5), (3, 6)]
```

Αν δώσουμε αυτή τη λίστα στη συνάρτηση dict, θα μας φτιάξει ένα λεξικό όπου το x θα είναι τα κλειδιά και το y θα είναι οι τιμές:

```
thedict = dict(zip(x,y))
print thedict
{ 1:4, 2:5, 3:6 }
```

Στον δικό μας κώδικα, το λεξικό ορίζεται κάπως έτσι:

```
self.forward_cipher = dict(zip(self.adjusted_letter, self.adjusted_
connections))
```

Και, σωστά παρατηρήσατε, ότι το έχουμε ονομάσει **forward_cipher**, αφού μπορούμε να το χρησιμοποιήσουμε μόνο για την ευθεία κρυπτογράφηση (αυτή που κινείται από το πληκτρολόγιο προς τον ανακλαστήρα). Μην ξεχνάτε ότι το λεξικό λειτουργεί προς τη μια φορά: τα κλειδιά είναι μοναδικά, οι τιμές όχι απαραίτητα!

Το καλό είναι ότι σε μας οι τιμές είναι επίσης μοναδικές! Έτσι, το μόνο που έχουμε να κάνουμε είναι να φτιάξουμε κι ένα αντίστροφο λεξικό, για την αντίστροφη κρυπτογράφηση (από τον ανακλαστήρα προς το lampboard):

```
self.reverse_cipher = dict(zip(self.adjusted_connections, self.adjusted_
letter))
```

Αν δείτε και τον τελικό μας κώδικα (στο enigma3.py, που θα κατεβάσετε από το http://bit.ly/dh023enigmasrc3), θα παρατηρήσετε τις απλές αλλαγές που έχουμε κάνει στις συναρτήσεις cipher και reflectCipher, ώστε να χρησιμοποιούν τα παραπάνω λεξικά.

Geheim!

Nicht ins Flugzeug mitnehmen!

Sonder-Maschinenschlüssel BGS 08 ❊

Datum	Walzenlage	Ringstellung	Steckerverbindungen	Kenngruppen
31.	I II V	10 14 02	BF SD AY HG OU QC WI RL XP ZK	yqv vuc xxo gvf
30.	V IV I	04 25 01	DI ZL RX UH QK PC VY GA SO EM	mqy vts gvt csx
29.	III V II	13 11 06	ZM BQ TP YX FK AR WH SO NJ DG	aky vdv oyo tzt
28.	I III II	09 16 12	NE MT RL OY HV IU GK FW PZ XC	nfh vco tur wnb
27.	III II I	06 03 15	BF GR SZ OM WQ TY HE JU XN KD	bec jmv vtp xdb
26.	I III V	19 26 08	GS VD CQ LE HI BO JP UZ FT RN	wvu yem buz rjk
25.	II I IV	05 01 16	KA ZH QP GR MF LJ OT EN BD YW	ktv muq cqm cpm
24.	III II IV	22 02 06	PI KM JB YU QS OV ZA GW CH XF	zcd iwo urp glg
23.	IV III II	08 11 07	SX TD QP HU FB YN CO IK WE GZ	epm mgz vqg vsm
22.	I V II	13 02 26	GP XH IW BO NU MD SA ZK QR LT	aam mvy jqq wqm
21.	IV I V	17 24 03	XC AQ OT UZ HD RG KM BL NS JW	ltl blu frk xrh
20.	IV I III	15 22 12	PO TV QC ZS EX WR BJ DK FU LA	non lic oxr usr
19.	V I III	13 24 21	HA GM DI VK JP YU EF TB ZL XQ	ecd ciq uvr ppt
18.	IV V I	23 09 20	XF PZ SQ GR AJ UO CN BV TM KI	fjh zts uqu cft
17.	III II V	21 24 15	UT ZC YN BE PK JX RS GF IA QH	oub eci pyf rqi
16.	IV III V	07 01 13	IN YJ SD UV GF BH TK QE AR OP	kex paw flw onw
15.	I IV II	15 04 25	TM IJ VK OY NX PR WL GA BU SF	sdr pbu byv khb
14.	III II IV	10 23 21	WT RE PC FY JA VD OI HK NX ZS	mhz lff lnq giy
13.	V I II	14 04 12	AN IV LH YP WM TR XU FO ZB ED	rqh ucm ldi ods
12.	II V I	07 19 02	HR NC IU DM TW GV FB ZL EQ OX	asy xza uvc fmr
11.	I V IV	13 15 11	NX EC RV GP SU DK IT FY BL AZ	gyd iuq ocb vef
10.	V II I	09 20 19	FN TA YJ SO EG PC VD KI XH WZ	pyz ace pru uyc
9.	I IV V	14 10 25	VK DW LH RF JS CX PT YB ZG MU	nyh fbd ohs jrp
8.	IV V I	22 04 16	PV XS ZU EQ BW CH AO RL JN TD	tck rts nro mkl
7.	V I IV	18 11 25	TS IK AV QP HW FM DX NG CY UE	mhw lwb mdm ybe
6.	IV I III	02 17 20	KZ FI WY MP DS HR CU XE QV NT	uwu vdk lrh mgd
5.	I V IV	26 09 14	VW LT PB FO ZK GS RI QJ HM XE	suw tsv nfp yjc
4.	IV III V	07 01 12	QS YA XW KR MP HT DU OV CL FZ	uby usi mhh mwb
3.	I II V	05 16 03	FW DL NX BV KM RZ HY IQ EC JU	tns von grw axl
2.	III I II	12 22 17	DW UO PY GR FS EQ KT CL AI ZB	smz lbl bkc sym
1.	I III II	04 18 06	ZN OM CR UI KP WQ SE JV LX TF	ghr vqv cya ayl

Αυτή η σελίδα υπήρχε σε κάθε υποβρύχιο των Γερμανών. Για κάθε μέρα περιέχει τις αρχικές ρυθμίσεις της μηχανής (τα είδη των rotors, τα ring settings (Ringstellung) και τις ρυθμίσεις του plugboard). Ο χειριστής πρέπει να περάσει αυτές τις ρυθμίσεις στη μηχανή για να λάβει το αρχικό μήνυμα το οποίο θα περιέχει, ουσιαστικά, το νέο μυστικό κλειδί για τις υπόλοιπες επικοινωνίες της ημέρας. Προφανώς, αν μια τέτοια σελίδα έπεφτε στα χέρια του εχθρού, η αποκρυπτογράφηση του Enigma θα ήταν παιχνιδάκι — τουλάχιστον για ένα μήνα. Για τον λόγο αυτό η εκτύπωση γινόταν με υδατοδιαλυτό μελάνι, ώστε ο χειριστής να το πετάξει στη θάλασσα αν αντιληφθεί ότι πρόκειται να συμβεί Κάτι Κακό (TM). Ο Alan Turing, χωρίς αυτό το χαρτί, πάλι κατάφερνε να σπάσει τον κώδικα της απόρθητης (κατά τους Γερμανούς) Enigma!

PlugBoard: Τα πράγματα σοβαρεύουν!

Για να φτιάξουμε μια πραγματικά στρατιωτικών προδιαγραφών μηχανή, χρειαζόμαστε το plugboard και δέκα καλώδια για τις αντίστοιχες συνδέσεις. Ευτυχώς, όλα αυτά είναι δωρεάν: Θα μας κοστίσουν μόνο μερικά ευχάριστα απογεύματα!

Θα πρέπει να θυμόμαστε ότι το plugboard λειτουργεί και προς τις δύο μεριές:

- Αν έχουμε βάλει ένα καλώδιο από το A↔K, όταν πιέζουμε το [A] στο πληκτρολόγιο το ρεύμα θα εξέρχεται προς τον δεξιό ρότορα, στην επαφή που αντιστοιχεί στο γράμμα «K». Αλλά κι όταν πιέζουμε το [K], το ρεύμα θα εξέρχεται από την επαφή «A»
- Στην αντίστροφη κρυπτογράφηση, όταν λαμβάνουμε το «K» στην έξοδο του δεξιού ρότορα, θα ανάβει το φωτάκι «A». Αλλά κι όταν λαμβάνουμε το «A» θα ανάβει το φωτάκι «K».

Προσέξτε λίγο την παγίδα: Το καλώδιο δεν λειτουργεί μόνο προς τη μια μεριά ως A→K αλλά κι ως K→A. Καταλαβαίνετε τώρα τη βαρύτητα του συμβολισμού A↔K.

Ωραία, αλλά πως θα τα γράψουμε όλα αυτά σε Python; Έχουμε πάρει φόρα, τώρα με τα dictionaries, άρα καλό θα είναι να τα χρησιμοποιήσουμε κι εδώ (παραλλαγή της γνωστής ρήσης «όταν το εργαλείο που κρατάς είναι σφυρί, τα πάντα μοιάζουν με καρφιά!»)

Ας μην ξεχνάμε ότι το plugboard είναι διάφανο: Αν δεν υπάρχει καλώδιο σε μια θέση, αυτή αντιστοιχίζεται στον εαυτό της. Ξεκινάμε λοιπόν φτιάχνοντας ένα λεξικό, στο οποίο κάθε γράμμα αντιστοιχίζεται στον εαυτό του:

```
self.letters = [chr(i) for i in range(65,91)]
self.translation_dict = dict(zip(self.letters, self.letters))
```

Στην init του Plugboard class θα δίνουμε ως παράμετρο ένα λεξικό plugs με τις αλλαγές που θέλουμε να υπάρχουν στην παραπάνω, προεπιλεγμένη αντιστοίχιση. Για παράδειγμα, δίνοντας

```
plugs = {'A':'F', 'B':'X', 'C':'L', 'O':'R', 'M':'U', 'D':'Y', 'I':'K', 'E':'N', 'G':'H', 'S':'W'}
```

εννοούμε την ύπαρξη καλωδίων που ενώνουν τα A↔F, B↔X, C↔L κ.ο.κ. Προσέξτε ότι αν και το λεξικό ως δομή υπονοεί, π.χ., μόνο A→F, εμείς θέλουμε και το F→A! Στην πραγματικότητα οι αλλαγές μας χρειάζονται και το παραπάνω λεξικό αντίστροφα, δηλαδή

```
{'F':'A', 'X':'B', 'L':'C', 'R':'O', 'U':'M', 'Y':'D', 'K':'I', 'N':'E', 'H': 'G', 'W':'S'}
```

Πιστεύω θα συμφωνήσετε μαζί μας πως είναι κάπως βαρετό (tedious!) να γράφουμε τα ίδια πράγματα δύο φορές. Έτσι, στην κλάση μας θα δημιουργήσουμε και το ανάστροφο λεξικό χρησιμοποιώντας μόνο το αρχικό plugs, καθώς και το αγαπημένο μας dict comprehension:

```
reverse_plugs = dict((value, key) for key,value in plugs.iteritems())
```

Όπως φαντάζεστε, το iteritems μάς επιτρέπει να διατρέχουμε τα ζεύγη «κλειδί: τιμή» (key, value) του λέξικου μας και, τοποθετώντας τα ανάποδα (value, key), να φτιάχνουμε το αντίστροφο λεξικό. Και πάλι, αυτό είναι δυνατόν γιατί στη περίπτωση μας οι τιμές είναι μοναδικές, οπότε μπορούν να χρησιμοποιηθούν κι ως κλειδιά.

Και να πως θα περάσουμε τις αλλαγές μας στο βασικό λεξικό self.translation_dict:

```
for i in plugs.iterkeys():
  self.translation_dict[i]=plugs[i]
```

Αντίστοιχα για το αντίστροφο λεξικό:

```
for i in reverse_plugs.iterkeys():
  self.translation_dict[i]=reverse_plugs[i]
```

Μετά από αυτά, είναι πολύ εύκολο να γράψουμε τη συνάρτηση cipher για το plugboard. Έχει μία και μοναδική γραμμή!

```
def cipher(self,letter):
  return self.translation_dict[letter]
```

Όλα τα παραπάνω μπορείτε να τα δείτε και να πειραματιστείτε στο αρχείο enigma3.py που έχετε κατεβάσει ήδη (ξέρετε, από το http://bit.ly/dh023enigmasrc3).

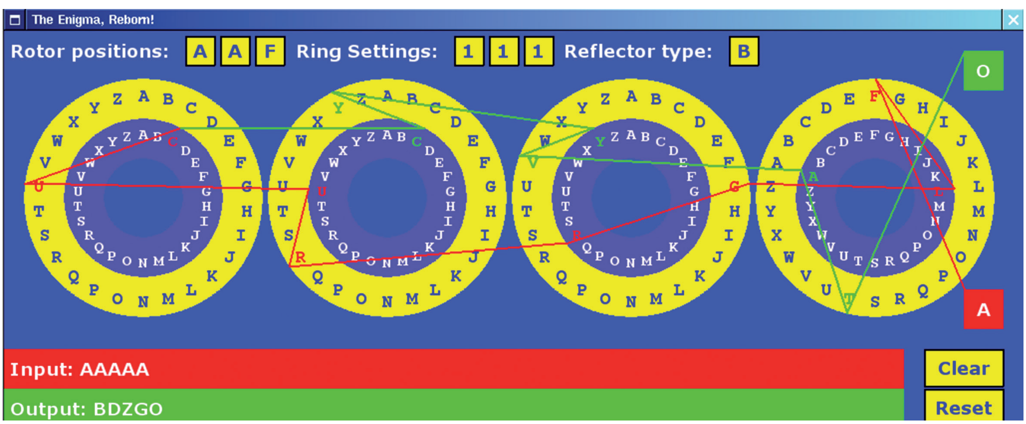
Η έκδοση γραφικών: Enigma Pygame

Μπορεί η enigma να ήταν ο φόβος και ο τρόμος της κρυπτογράφησης στο Β' Παγκόσμιο πόλεμο, όμως πλέον την έχουμε μετατρέψει σε παιχνίδι! Στα πλαίσια αυτής τη μετατροπής, θεωρήσαμε σκόπιμο να τη γράψουμε στην αγαπημένη μας βιβλιοθήκη, την PyGame. Μπορείτε να κατεβάσετε από τώρα την αρχική εκδοχή του προγράμματος enigma-pygame.py από http://bit.ly/dh023enigmasrc3 και από το επόμενο τεύχος θα εξηγήσουμε τη λειτουργία του, θα περάσουμε τα κομμάτια του Enigma που λείπουν και θα ολοκληρώσουμε αυτή τη σειρά άρθρων.

Για να εκτελέσετε την εκδοχή γραφικών θα πρέπει να κατεβάσετε και να εγκαταστήσετε στο λειτουργικό σας την βιβλιοθήκη PyGame. Μπορείτε να βρείτε λεπτομέρειες σε αυτή τη σελίδα:

    http://pygame.org/download.shtml

Εννοείται ότι αν έχετε ακολουθήσει τη σειρά άρθρων μας για το PyGame, τότε δεν χρειάζεται να εγκαταστήσετε κάτι παραπάνω. Σας αφήνουμε τώρα να πειραματιστείτε πάνω στην πλήρη μηχανή μας κι ανανεώνουμε το —γραφικό— ραντεβού μας για το επόμενο τεύχος. Να έχετε πολλά ευχάριστα (και δροσερά) απογεύματα!

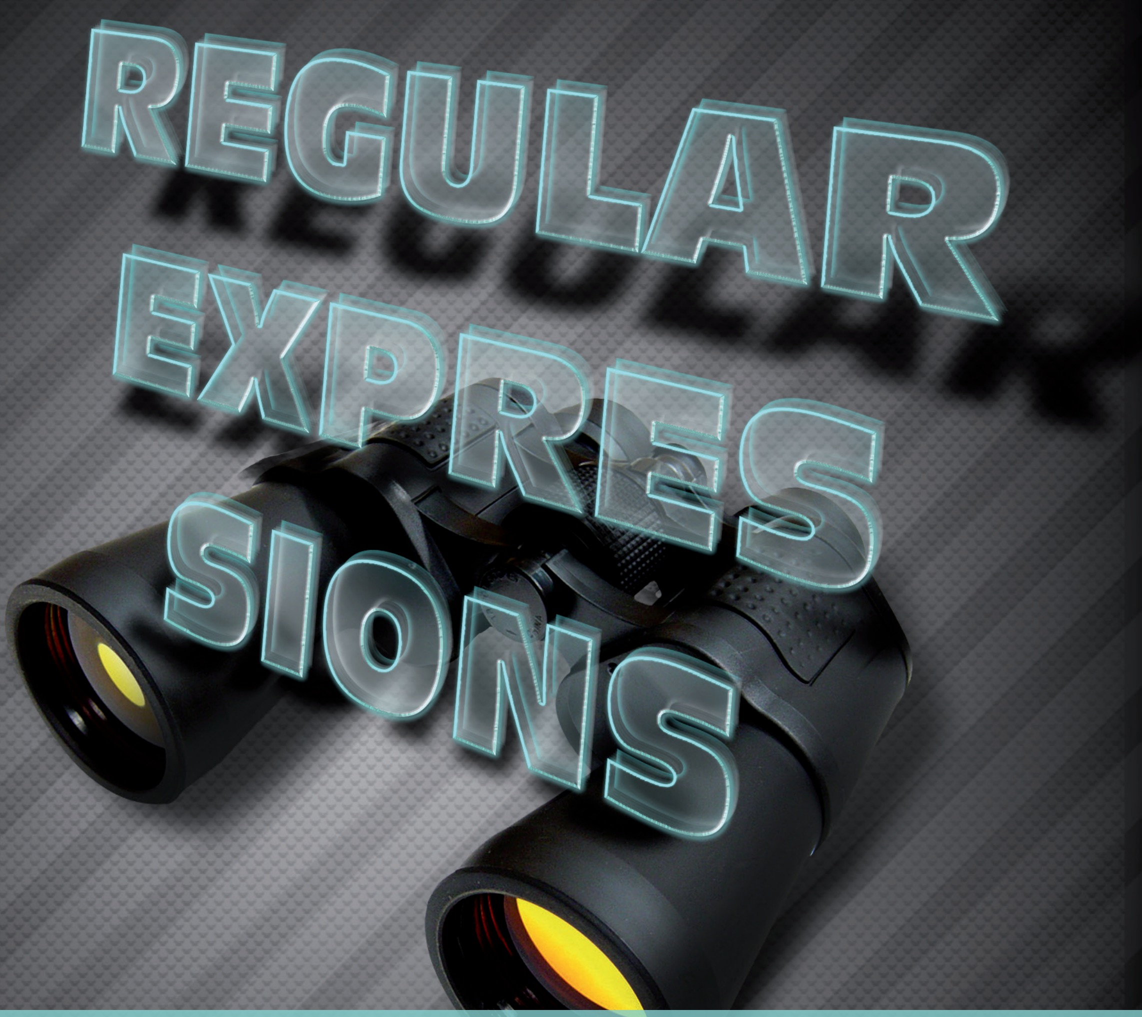


Η εξομοίωσή μας σε PyGame, που μπορείτε να χρησιμοποιήσετε ήδη από αυτό το τεύχος. Το περιβάλλον έχει βασιστεί στην πολύ καλή εξομοίωση του enigmaco.de, αλλά καθώς είναι γραμμένο σε Python κι όχι σε Flash δεν θα χρειαστείτε 256 πυρήνες για να το τρέξετε! Καθώς πιέζετε γράμματα, οι γραμμές δείχνουν την πορεία της κρυπτογράφησης μέσα στους ρότορες. Περισσότερα για τα Μαθηματικά της σχεδίασης θα δούμε στο επόμενο τεύχος. Στον κώδικα, όμως, θα δείτε ότι όλη η προεργασία που κάναμε με την text based εκδοχή του παιχν... της μηχανής, δεν πήγε χαμένη: Στις κλάσεις μας έχουμε απλά προσθέσει συναρτήσεις για τη γραφική σχεδίαση.

Skill: Intermediate
Tags: regular expressions, lookahead, lookbehind, greedy, lazy

by Spir@lEvolution

REGULAR EXPRESSIONS



Οι πανίσχυρες κανονικές εκφράσεις [μέρος 3]

Μέχρι στιγμής γνωρίζουμε όλα τα βασικά συστατικά καθώς και μερικά συντακτικά σχήματα που επιτρέπουν τη δημιουργία ιδιαίτερα περιγραφικών εκφράσεων. Για το τέλος αφήσαμε μια λειτουργία που διευρύνει περαιτέρω την περιγραφική ικανότητα των κανονικών εκφράσεων, καθώς κι ορισμένες λεπτομέρειες που οφείλουμε να γνωρίζουμε ώστε να σχηματίζουμε σαφείς κι εύστοχες εκφράσεις...

Στο προηγούμενο άρθρο της σειράς αναφερθήκαμε στα alternation (εναλλαγή) και eagerness (ενθουσιασμός;) που διέπουν τη λειτουργία ορισμένων μηχανών κανονικών εκφράσεων. Μελετήσαμε επίσης και τα σημεία αναφοράς (anchors), τα οποία δεν περιγράφουν τη σύνθεση ενός string —δηλαδή τους χαρακτήρες που το απαρτίζουν— αλλά μια θέση μέσα σ' αυτό. Στο τέλος γνωρίσαμε και τις λεγόμενες αναφορές (backreferences), οι οποίες επιτρέπουν να αναφερόμαστε σε συγκεκριμένα τμήματα ενός string τα οποία έχουν ήδη περιγραφεί *και* εντοπιστεί από προηγούμενα τμήματα της εκάστοτε κανονικής έκφρασης. Αυτή τη φορά θα μελετήσουμε έναν ακόμα μηχανισμό που θα διευρύνει την εκφραστική μας δύναμη, ενώ θα αναφερθούμε και σε μερικές ιδιαιτερότητες που διέπουν τη λειτουργία των μηχανών κανονικών εκφράσεων. Με τα εφόδια που έχουμε ήδη αποκτήσει, καθώς κι όλα όσα θα γνωρίσουμε, θα είμαστε σε θέση να συντάσσουμε σύντομες και ιδιαίτερα «εύστοχες» κανονικές εκφράσεις!

Έλεγχος ύπαρξης

Στην έκδοση 5 της γλώσσας Perl εισήχθησαν δύο μηχανισμοί που ονομάζονται lookahead και lookbehind κι αναφέρονται ως «lookarounds». Με τη βοήθειά τους μπορούμε να περιγράψουμε ένα string και να επιβεβαιώσουμε/εξασφαλίσουμε ότι βρίσκεται πριν (lookbehind) ή μετά (lookahead) από όσα περιγράφει η υπόλοιπη κανονική έκφραση. Επιπρόσθετα, με τους ίδιους μηχανισμούς μπορούμε να επιβεβαιώσουμε/εξασφαλίσουμε ότι ένα string *δεν* βρίσκεται πριν (negative lookbehind) ή μετά (negative lookahead) από όσα περιγράφει η υπόλοιπη κανονική έκφραση. Αλλά σ' αυτό το σημείο ενδέχεται να 'χετε μπερδευτεί από τις αφηρημένες περιγραφές. Ας δούμε μερικά παραδείγματα και θα κατανοήσετε αμέσως τη λειτουργία και τη χρησιμότητα των εν λόγω μηχανισμών.

- **Lookahead.** Ας υποθέσουμε ότι θέλουμε να δούμε όλες τις επιλογές που έχουν ενεργοποιηθεί στο αρχείο ρυθμίσεων του SSH server του συστήματός μας. Ουσιαστικά, αυτό που ψάχνουμε είναι όλες οι γραμμές του αρχείου /etc/ssh/sshd_config που τελειώνουν με το « yes». Από αυτές τις γραμμές, όμως, θέλουμε μόνο την εκάστοτε επιλογή! Επομένως, αναζητάμε τις λέξεις εκείνες που βρίσκονται πριν από το « yes» κι όχι το ίδιο το « yes». Για να πετύχουμε κάτι τέτοιο, αρκεί να χρησιμοποιήσουμε το μηχανισμό lookahead ως εξής:

```
$ grep -P --color=auto ".*(?= yes$)" /etc/ssh/sshd_config
UsePrivilegeSeparation yes
PermitRootLogin yes
StrictModes yes
RSAAuthentication yes
PubkeyAuthentication yes
IgnoreRhosts yes
#IgnoreUserKnownHosts yes
#PasswordAuthentication yes
```

```
#KerberosOrLocalPasswd yes
#KerberosTicketCleanup yes
#GSSAPICleanupCredentials yes
X11Forwarding yes
PrintLastLog yes
TCPKeepAlive yes
UsePAM yes
```

Όπως βλέπετε, το string που θέλουμε να επιβεβαιώσουμε ότι έπεται όσων αναζητάμε (το περιεχόμενου του lookahed) τοποθετείται μέσα σε παρενθέσεις. Συγκεκριμένα, μετά από την αριστερή παρένθεση τοποθετούμε το λατινικό ερωτηματικό, στη συνέχεια το χαρακτήρα της ισότητας (ίσον) και αμέσως μετά το string που θέλουμε να τσεκάρουμε αν εμφανίζεται. Η γενική μορφή των lookahead είναι η ακόλουθη:

```
(?=regex)
```

Στο παράδειγμά μας, στη θέση του regex βρίσκεται η έκφραση « yes$». Προφανώς, αυτή η έκφραση περιγράφει το string « yes», το οποίο όμως βρίσκεται στο τέλος της γραμμής (αυτό εξασφαλίζεται με το σημείο αναφοράς που συμβολίζει το τέλος του string ή της γραμμής, δηλαδή τον χαρακτήρα $). Τελικά, με την κανονική έκφραση που δίνουμε στο grep, ζητάμε όλους τους χαρακτήρες οι οποίοι ακολουθούνται από ένα « yes» και τον χαρακτήρα τερματισμού της γραμμής! Η ίδια αναζήτηση θα μπορούσε να πραγματοποιηθεί και χωρίς το lookahead, ως εξής:

```
$ grep -P --color=auto ".* yes$" /etc/ssh/sshd_config
UsePrivilegeSeparation yes
PermitRootLogin yes
StrictModes yes
RSAAuthentication yes
PubkeyAuthentication yes
IgnoreRhosts yes
#IgnoreUserKnownHosts yes
#PasswordAuthentication yes
#KerberosOrLocalPasswd yes
#KerberosTicketCleanup yes
```

```
#GSSAPICleanupCredentials yes
X11Forwarding yes
PrintLastLog yes
TCPKeepAlive yes
UsePAM yes
```

Παρατηρήστε ότι σε αυτή την περίπτωση, τα αποτελέσματα της αναζήτησης περιλαμβάνουν και το τελικό « yes» κάθε γραμμής. Αυτό μπορεί να μην μας ενοχλεί καθόλου στο συγκεκριμένο παράδειγμα, αλλά υπάρχουν και περιπτώσεις που θα ήταν ιδιαίτερα ενοχλητικό.

- **Negative Lookahead.** Έστω ότι ψάχνουμε την τελευταία εμφάνιση μιας λέξης σε μια πρόταση. Εντάξει, το ομολογούμε: Αυτό το παράδειγμα είναι ελάχιστα χρήσιμο, αλλά θα αναδείξει με τον καλύτερο τρόπο τη χρησιμότητα του negative lookahead ;) Ας υποθέσουμε ότι θέλουμε να εντοπίσουμε την τελευταία εμφάνιση της λέξης «thing» μέσα στην ακόλουθη πρόταση: *«The major difference between a thing that might go wrong and a thing that cannot possibly go wrong is that when a thing that cannot possibly go wrong goes wrong it usually turns out to be impossible to get at or repair.»* (Πρόκειται για ένα απόφθεγμα του Douglas Adams, στο χιούμορ του οποίου ο αρθρογράφος τρέφει ιδιαίτερη εκτίμηση.) Ας υποθέσουμε επίσης ότι έχουμε τοποθετήσει αυτή την πρόταση στο αρχείο quote.txt. Όπως είπαμε, αυτό που θέλουμε να πετύχουμε είναι ο εντοπισμός της τελευταίας εμφάνισης της λέξης thing. Με άλλα λόγια, θέλουμε να βρούμε το string «thing», το οποίο όμως *δεν ακολουθείται* από το string «thing»! Αυτό μπορεί να γίνει ως εξής:

```
$ grep -P --color=auto 'thing(?!.*thing)' quote.txt

The major difference between a thing that might go wrong and a
thing that cannot possibly go wrong is that when a thing that
cannot possibly go wrong goes wrong it usually turns out to be
impossible to get at or repair.
```

Το string που *δεν θέλουμε* να ακολουθεί εκείνο που αναζητάμε, τοποθετείται και πάλι μέσα σε παρενθέσεις. Αυτή τη φορά όμως, μετά από την αριστερή παρένθεση ακολουθεί ένα λατινικό ερωτηματικό και ένα θαυμαστικό. Η σύνταξη του negative lookahead διαφοροποιείται από εκείνη του (positive) lookahead ως προς έναν χαρακτήρα: Στη θέση του συμβόλου της ισότητας (ίσον) χρησιμοποιείται ένα θαυμαστικό. Η γενική μορφή των negative lookahead φαίνεται παρακάτω:

```
(?!regex)
```

Στο παράδειγμά μας, μέσα στο negative lookahead τοποθετήσαμε την έκφραση «.*thing». Δηλαδή, τοποθετήσαμε μια κανονική έκφραση που περιγράφει ένα string με οποιουσδήποτε και οσουσδήποτε χαρακτήρες και το οποίο τε-

λειώνει με τη λέξη thing. Έτσι, η μηχανή των κανονικών εκφράσεων θα αναζητήσει μια εμφάνιση της λέξης thing, η οποία όμως *δεν ακολουθείται* από κάποιο string που τελειώνει με τη λέξη thing! Απλούστατο… ;)

- **Lookbehind.** Ας υποθέσουμε τώρα ότι θέλουμε να εντοπίσουμε όλα τα σχόλια μέσα στον πηγαίο κώδικα ενός προγράμματος σε C, τα οποία περιλαμβάνουν μια συγκεκριμένη λέξη. Για παράδειγμα, τα σχόλια που περιέχουν τη λέξη sample. Αυτό μπορεί να γίνει πανεύκολα ως εξής:

```
$ grep -P --color=auto '(?<=//).*sample.*' code.txt
volatile byte sample;            // new sample
volatile byte sample_cnt;        // sample counter for delays
// Will be used for sample generation
```

Εδώ χρησιμοποιούμε το lookbehind για να εντοπίσουμε τα strings που περιέχουν τη λέξη sample και τα οποία έπονται των χαρακτήρων «//», οι οποίοι σηματοδοτούν την έναρξη ενός σχολίου. Η σύνταξη του lookbehind ομοιάζει με εκείνη του lookahead. Η μόνη διαφορά έγκειται στην προσθήκη του συμβόλου «μικρότερο από» (<) ανάμεσα στο λατινικό ερωτηματικό και στο χαρακτήρα της ισότητας. Παρακάτω, βλέπουμε τη γενική μορφή ενός lookbehind:

```
(?<=regex)
```

Όπως και στην περίπτωση του lookahead, η ίδια αναζήτηση θα μπορούσε να πραγματοποιηθεί και με έναν απλούστερο τρόπο:

```
$ grep -P --color=auto '//.*sample.*' code.txt
```

Ωστόσο, σε αυτή την περίπτωση τα αποτελέσματα θα περιλάμβαναν μαζί με τα σχόλια και τους χαρακτήρες «//».

- **Negative lookbehind.** Ας υποθέσουμε τώρα ότι θέλουμε να απομονώσουμε όλους τους αριθμούς που εμφανίζονται στον πηγαίο κώδικα ενός προγράμματος και οι οποίοι δεν αποτελούν δείκτες σε κάποιον πίνακα, ούτε τμήμα της ονομασίας κάποιας μεταβλητής. Με άλλα λόγια, θέλουμε όλους τους αριθμούς που δεν βρίσκονται αμέσως μετά από μία αριστερή αγκύλη (δεν αποτελούν δείκτες σε πίνακα), ούτε έχουν ακριβώς στα αριστερά τους κάποιο γράμμα (δεν αποτελούν τμήμα της ονομασίας κάποιας μεταβλητής[1]). Τελικά, θέλουμε όλους τους αριθμούς για τους οποίους ικανοποιούνται οι δύο παραπάνω συνθήκες. Επειδή αυτές οι συνθήκες περιλαμβάνουν μια άρνηση (*δεν* θέλουμε να εμφανίζεται κάτι) κι επειδή τσεκάρουν κάτι που βρίσκεται *πριν* από το ζητούμενο (τους αριθμούς) θα στηριχτούμε στο negative lookback:

```
$ grep -P --color=auto '(?<!\[)(?<!\w)(\d*)' code.txt
```

1. Τα ονόματα των μεταβλητών δεν επιτρέπεται να ξεκινούν με αριθμητικά ψηφία. Τουλάχιστον όχι στη C και σε πολλές άλλες γλώσσες. Επομένως, δεν χρειάζεται να ελέγξουμε αν υπάρχουν γράμματα στα δεξιά του αριθμού.

Η σύνταξη των negative lookback μοιάζει με εκείνη των negative lookahead, μόνο που ανάμεσα στο ερωτηματικό και στο θαυμαστικό τοποθετείται ο χαρακτήρας «μικρότερο από» (<):

```
(?<!regex)
```

Μελετήστε λίγο το παράδειγμα. Όπως θα δείτε, για να εξασφαλίσουμε ότι ικανοποιούνται οι *δύο* συνθήκες, συμπεριλάβαμε *δύο* negative lookbacks — ένα για κάθε συνθήκη! Πριν προχωρήσουμε, σημειώστε ότι ο ειδικός χαρακτήρας \w αναφέρεται σε όλους τους χαρακτήρες που μπορεί να περιλαμβάνει μια λέξη, ενώ ο ειδικός χαρακτήρας \d συμβολίζει όλα τα αριθμητικά ψηφία. Ουσιαστικά, αυτοί οι ειδικοί χαρακτήρες αποτελούν συντομεύσεις για τις κλάσεις [a-zA-Z0-9_] και [0-9] αντίστοιχα. Μετά από τα δύο negative lookback ακολουθεί η περιγραφή ενός αριθμού με οσαδήποτε ψηφία. Παρατηρείστε ότι αυτό το κομμάτι της έκφρασης το έχουμε κλείσει σε παρενθέσεις. Αναρωτιέστε γιατί; Σε αυτό το σημείο χρειάζεται λίγη παραπάνω προσοχή: Χωρίς τις παρενθέσεις, όταν το grep θα συναντούσε κάποιον αριθμό μέσα στο όνομα μιας μεταβλητής ή μέσα σε αγκύλες, θα αγνοούσε μόνο το πρώτο ψηφίο του αριθμού. Βλέπετε, το δεύτερο ψηφίο δεν θα είχε στα αριστερά του κάποιον χαρακτήρα λέξης, ούτε την αριστερή αγκύλη. Με τη χρήση των παρενθέσεων εξασφαλίζουμε ότι οι δύο συνθήκες που ελέγχουμε με τα negative lookback αναφέρονται στο σύνολο των αριθμητικών ψηφίων του εκάστοτε αριθμού κι όχι μόνο στο πρώτο του ψηφίο!

Σημεία προσοχής

Κατ' αρχάς, πρέπει να ξεκαθαρίσουμε ότι τα παραδείγματα που μελετήσαμε, ανεξάρτητα από το αν και κατά πόσο είναι χρήσιμα, είχαν κυρίως μία αποστολή: Να παρουσιάσουν τη σύνταξη των lookaround. Είναι πολύ σημαντικό να έχουμε κατά νου ότι οι εν λόγω μηχανισμοί δεν προσδιορίζουν κάποια θέση! Τα lookaround εξασφαλίζουν μόνο την ύπαρξη ή την απουσία κάποιων strings, μέσα στο ευρύτερο string. Μπορείτε να φαντάζεστε τα lookaround σαν λογικούς ελέγχους (if) των οποίων η έκβαση (ναι ή όχι — 1 ή 0) καθορίζουν το αν η μηχανή κανονικών εκφράσεων θα παρουσιάσει κάποιο αποτέλεσμα ή όχι. Υποσχόμαστε να επανέλθουμε σε επόμενο άρθρο του περιοδικού, όπου και θα παρουσιάσουμε χρήσιμα παραδείγματα των κανονικών εκφράσεων. Τέλος, πρέπει να σημειώσουμε ότι ειδικά στην περίπτωση των lookbehind (των negative αλλά και των positive), δεν μπορούμε να δώσουμε string με απροσδιόριστο μήκος. Αυτό σημαίνει ότι μέσα σε ένα lookback δεν μπορούμε να συμπεριλάβουμε τους χαρακτήρες επανάληψης (*, + και ?). Αυτός ο περιορισμός ισχύει για σχεδόν όλες τις μηχανές κανονικών εκφράσεων, εκτός από εκείνη της Perl και του .NET.

Αχόρταγες μηχανές

Στο προηγούμενο άρθρο της σειράς αναφέραμε ότι οι περισσότερες μηχανές κανονικών εκφράσεων είναι ενθουσιώδεις (eager) και βιάζονται να εμφανίσουν κάποιο αποτέλεσμα. Αυτή η συμπεριφορά επηρεάζει το χειρισμό των εναλλακτικών επιλογών μέσα σε ένα alternation (regex_1 | regex_2 | … | regex_n). Έτσι, οι διάφορες μηχανές παρουσιάζουν σαν αποτέλεσμα την πρώτη επιλογή που συναντούν σε ένα alternation και ικανοποιείται, χωρίς να ελέγξουν αν υπάρχει κάποια καλύτερη!

Εκτός από αυτή την ιδιοτροπία, οι μηχανές κανονικών εκφράσεων έχουν και μία ακόμα, αρκετά ύπουλη: Είναι αχόρταγες (greedy)! Ας δούμε ένα παράδειγμα.

Ας υποθέσουμε ότι θέλουμε να απομονώσουμε όλα τα strings που περικλείονται από εισαγωγικά, μέσα σε ένα κείμενο. Ας υποθέσουμε, επίσης, ότι το κείμενο στο οποίο θα πραγματοποιήσουμε την αναζήτηση είναι το ακόλουθο:

```
This is a line with "a string" in quotes.

This is another line with "a string" in quotes an "another one", also
in quotes.

One more line with "string 1", "string 2" and "string 3", all in quotes.
```

Η αναζήτησή μας θα μπορούσε να γίνει κάπως έτσι:

```
$ grep -P --color=auto ".*" text.txt

This is a line with "a string" in quotes.

This is another line with "a string" in quotes an "another one", also
in quotes.

One more line with "string 1", "string 2" and "string 3", all in quotes.
```

Βλέπετε το λάθος; Στην πρώτη γραμμή, η αναζήτηση λειτούργησε ακριβώς όπως αναμέναμε και απομόνωσε το string με τα εισαγωγικά. Στη δεύτερη και στην τρίτη γραμμή, όμως, τα επιμέρους strings αντιμετωπίστηκαν ως ένα συνεχόμενο. Σε αυτές τις γραμμές, η μηχανή εντόπισε από ένα string, το οποίο ξεκινά με την πρώτη εμφάνιση των εισαγωγικών και τελειώνει με την τελευταία τους εμφάνιση! Αυτή η συμπεριφορά οφείλεται στην αχόρταγη φύση της μηχανής των κανονικών εκφράσεων.

Όπως γνωρίζουμε ήδη, ο χαρακτήρας της τελείας λειτουργεί σαν μπαλαντέρ και αναφέρεται σε οποιονδήποτε άλλο χαρακτήρα (ακόμα και στα εισαγωγικά). Ο αστερίσκος αποτελεί έναν χαρακτήρα επανάληψης που περιγράφει *οσεσδήποτε* εμφανίσεις του αμέσως προηγούμενου χαρακτήρα και εν προκειμένω της τελείας. Έτσι, με την κανονική έκφραση που έχουμε δώσει στο grep, περιγράφουμε ένα string οποιουδήποτε μεγέθους και οποιασδήποτε σύνθεσης, το οποίο περικλείεται σε εισαγωγικά. Όμως η μηχανή των κανονικών εκφράσεων είναι αχόρταγη κι έτσι παρουσιάζει σαν αποτέλεσμα το *μεγαλύτερο* string, που ικανοποιεί αυτή την περιγραφή!

Πρόχειρη λύση

Το σενάριο που μελετήσαμε παραπάνω κρύβει δύο παγίδες: Η μία σχετίζεται με τη φύση των μηχανών των κανονικών εκφράσεων, ενώ η δεύτερη σχετίζεται με την ερμηνεία του ειδικού χαρακτήρα της τελείας. Βλέπετε, η τελεία μπορεί να περιγράψει *οποιονδήποτε* χαρακτήρα κι ενώ αυτό μοιάζει να είναι χρήσιμο (στο κάτω κάτω, για χαρακτήρα-μπαλαντέρ πρόκειται), είναι εύκολο να δημιουργήσει και προβλήματα. Στη συγκεκριμένη περίπτωση, το πρόβλημα ξεκινά από το γεγονός ότι η τελεία περιγράφει *και* τον χαρακτήρα των εισαγωγικών. Μετά μπαίνει στο παιχνίδι η αδηφάγος όρεξη της μηχανής των κανονικών εκφράσεων, με αποτέλεσμα την προβληματική συμπεριφορά που είδαμε προηγουμένως. Μια πρόχειρη λύση στο πρό-

βλημα θα ήταν να αντικαταστήσουμε την τελεία με μία κλάση, η οποία περιλαμβάνει όλους τους χαρακτήρες, εκτός από εκείνο των εισαγωγικών:

```
$ grep -P --color=auto "[^"]*?" text.txt

This is a line with "a string" in quotes.

This is another line with "a string" in quotes an "another one", also
in quotes.

One more line with "string 1", "string 2" and "string 3", all in quotes.
```

Τώρα, οι χαρακτήρες των εισαγωγικών που συναντά η μηχανή ταυτίζονται είτε με τον αρχικό, είτε με τον τελικό χαρακτήρα της κανονικής έκφρασης. Έτσι, η μηχανή καταφέρνει να απομονώσει όλα τα επιμέρους strings με επιτυχία!

Επιβάλλοντας τη λιτότητα

Η λύση που μόλις παρουσιάσαμε λειτουργεί άψογα, αλλά μόνο στο συγκεκριμένο παράδειγμα. Υπάρχουν περιπτώσεις που θα ήταν δύσκολο κι ενδεχομένως χρονοβόρο να σχηματίσουμε τις κατάλληλες κλάσεις, οι οποίες θα αντικαθιστούσαν το χαρακτήρα του μπαλαντέρ με επιτυχία. Εξάλλου, αν κάθε φορά που θέλουμε να χρησιμοποιήσουμε το μπαλαντέρ καταφεύγουμε στη συγκρότηση μιας κλάσης χαρακτήρων, η τελική κανονική έκφραση θα είναι μακροσκελής και δυσανάγνωστη. Για αυτούς τους λόγους, οι μηχανές κανονικών εκφράσεων επιτρέπουν στο χρήστη να αλλάξει τη συμπεριφορά τους από greedy σε lazy! Με απλά λόγια, μπορούν να μετατραπούν από αχόρταγες σε... λιτοδίαιτες. Για να πετύχουμε κάτι τέτοιο, αρκεί να τοποθετήσουμε ένα λατινικό ερωτηματικό ακριβώς δίπλα από τον εκάστοτε χαρακτήρα των επαναλήψεων (τον αστερίσκο ή το σύμβολο της πρόσθεσης). Αξιοποιώντας αυτή τη δυνατότητα, το πρόβλημα που παρουσιάσαμε παραπάνω λύνεται ως εξής:

```
$ grep -P --color=auto ".*?" text.txt

This is a line with "a string" in quotes.

This is another line with "a string" in quotes an "another one", also
in quotes.

One more line with "string 1", "string 2" and "string 3", all in
quotes.
```

Όπως βλέπετε, με τη χρήση του λατινικού ερωτηματικού αμέσως μετά τον αστερίσκο, η αναζήτηση επέστρεψε ακριβώς αυτό που περιμέναμε: Κάθε επιμέρους string ξεχωριστά! Βλέπετε, το λατινικό ερωτηματικό επιβάλει στη μηχανή των κανονικών εκφράσεων να μην αναζητά το μεγαλύτερο δυνατό string (greedy) που αρχίζει και τελειώνει με εισαγωγικά, αλλά το μικρότερο δυνατό (lazy)!

Ασχοληθείτε και πειραματιστείτε με τα regular expressions. Σύντομα, μέσα από τις σελίδες του περιοδικού, θα δείτε ότι έχουν πολλές περισσότερες εφαρμογές από τις πολύπλοκες αναζητήσεις ;)

Skill: Advanced
Tags: ZTE, Blade, Android, ROM, root, ClockWork, CyanogenMod



Ακονίστε τη λεπίδα της ZTE

Είναι πολλές οι περιπτώσεις κατά τις οποίες κακές επιλογές του τμήματος marketing έθεσαν αξιόλογα προϊόντα εκτός αγοράς. Κάτι παρόμοιο συμβαίνει και με το after sales service, φυσικά. Ευτυχώς για εμάς, υπάρχουν κοινότητες προγραμματιστών με τη βοήθεια των οποίων τεχνολογικά προϊόντα που μοιάζουν ξεπερασμένα μπορούν να αποκτήσουν χρησιμότητα εκ νέου. Μια απ' αυτές τις αδικοχαμένες συσκευές είναι το Blade της ZTE. Ε, λοιπόν, σε αυτό το γερασμένο τηλέφωνο μπορούμε να εγκαταστήσουμε μια από τις τελευταίες εκδόσεις του Android!

Τα τελευταία χρόνια έχουν έρθει τα πάνω κάτω στο χώρο της κινητής τηλεφωνίας. Το ποσοστό που κατείχαν στην αγορά οι πάλαι ποτέ κραταιοί κατασκευαστές κατακρημνίστηκε, οι ισορροπίες άλλαξαν και στο χώρο εισήλθαν πολλές νέες εταιρείες. Οι Κινέζοι κατασκευαστές, με την τεράστια τεχνογνωσία[1] και την επιθετική τιμολογιακή πολιτική, δεν θα μπορούσαν να λείψουν από την αγορά. Ανάμεσα σε αυτές τις εταιρείες ξεχώρισαν δύο, για τα (σχετικά) ποιοτικά τους προϊόντα. Μιλάμε για τις Huawei και ZTE. Για την δεύτερη ειδικά, μπορούμε να πούμε ότι αυτό που την καθιέρωσε στην αγορά ήταν μια πολύ συγκεκριμένη συσκευή. Μιλάμε για το ZTE Blade, που κυκλοφόρησε πριν τριάμισι χρόνια, με οθόνη 3,5 ιντσών, Android και γενικότερα πολύ υψηλό δείκτη value for money.

The rise and fall...

Το ZTE Blade είχε τιμολογηθεί κάπου στα 150 ευρώ, ενώ οι ανταγωνιστικές προτάσεις κόστιζαν τουλάχιστον τη διπλάσια τιμή. Η συσκευή διέθετε επεξεργαστή ARMv6 στα 600MHz και 512MB μνήμης RAM, η μισή από την οποία ήταν αόρατη στο χρήστη εξαιτίας του firmware που είχε εγκαταστήσει ο εγχώριος εισαγωγέας (η Wind). Η επιλογή της εταιρίας να εισαγάγει τη συγκεκριμένη συσκευή ήταν μάλλον ακατανόητη, αφού και η κατασκευάστρια εταιρεία δεν είχε ακόμα αντιπροσωπεία στην Ελλάδα. Ωστόσο, τίποτα από όλα αυτά δεν πτόησε τους αγοραστές που αναζητούσαν μια καλή εμπειρία Android, συνδυασμένη με μια πολύ καλή τιμή — τουλάχιστον για τα δεδομένα της εποχής. Επιπλέον, όσοι χρήστες είχαν το μικρόβιο του σκαλίσματος και των επεμβάσεων, είχαν βρει τρόπο να ξεπεράσουν τον ανόητο περιορισμό της μνήμης και μ' αυτόν τον τρόπο είχαν αναβαθμίσει τη συσκευή τους δωρεάν!

Τα χρόνια πέρασαν, η ZTE κυκλοφόρησε τα Blade II και III με αναβαθμισμένα χαρακτηριστικά, ενώ το Android έφτασε αισίως στην έκδοση Jelly Bean (4.2). Η τελευταία διαθέσιμη αναβάθμιση για το Blade αφορούσε μόλις στην έκδοση του Android 2.2 (Froyo). Μάλιστα, η εγκατάσταση αυτής της αναβάθμισης εγκυμονούσε τον κίνδυνο να "χαθεί" εκ νέου η μισή μνήμη! Έτσι, οι περισσότεροι κάτοχοι του πρώτου Blade παρέμειναν με το Android 2.1 (Eclair). Από τότε, το Blade άρχισε σιγά-σιγά να παραγκωνίζεται και να καταλήγει παροπλισμένο σε ντουλάπια και συρτάρια.

...and the new rise ;)

Αποφασίσαμε να διατηρήσουμε στη ζωή την πρώτη «λεπίδα» της ZTE και μάλιστα να την έχουμε όσο πιο αστραφτερή γίνεται! Αν είσαστε κι εσείς ένας από τους τυχερούς κατόχους της, συνεχίστε την ανάγνωση του άρθρου. Αν όχι, συνεχίστε και πάλι, αφού όλο και κάποιος γνωστός σας μπορεί να την έχει και να χρειάζεται τη βοήθειά σας ή, ακόμα καλύτερα, να έχει αποφασίσει ότι δεν την χρειάζεται και να σας τη χαρίσει.

Για να επιτύχουμε τον στόχο μας, που δεν είναι άλλος από το να αναβαθμίσουμε το ZTE Blade με μια πολύ πιο πρόσφατη έκδοση του Android, θα καταφύγουμε στη χρήση ενός custom ROM. Μιλάμε για τροποποιημένες εκδόσεις του λειτουργικού συστήματος Android, οι οποίες έχουν δημιουργηθεί από άλλους χρήστες κι όχι από την κατασκευάστρια εταιρεία. Βέβαια, κάπου εδώ πρέπει να σας προειδοποιήσουμε ότι με όλα τα ανεπίσημα updates ακυρώνεται η όποια εγγύηση, ενώ την ευθύνη για τυχόν μπρικάρισμα[2] ή δυσλειτουργία της συσκευής τη φέρει ο ίδιος ο χρήστης.

1. Είμαστε σίγουροι πως γνωρίζετε ότι σχεδόν όλα τα τεχνολογικά προϊόντα (ακόμα και τα Designed in California) κατασκευάζονται με τη μέθοδο του outsourcing, σε εργοστάσια στην Κίνα.
2. Δημιουργία συσκευής-τούβλου, που ουσιαστικά δεν μπορεί να εκκινήσει τη λειτουργία της.

Επιλογή ROM

Για πολύ καιρό, ο επεξεργαστής του Blade και συγκεκριμένα η απουσία κατάλληλων βιβλιοθηκών, απέτρεπαν τη λειτουργία του hardware video acceleration. Αυτό το χαρακτηριστικό, όμως, είναι κρίσιμο για τη σωστή λειτουργία των πρόσφατων εκδόσεων του Android (από 4 και πάνω).

Ευτυχώς, μετά την εύρεση κατάλληλης λύσης μπορούμε να έχουμε σταθερά και λειτουργικότατα ROM, με σχετικά πρόσφατες εκδόσεις του Android. Εμείς επιλέξαμε το Eco CM9. Πρόκειται για ένα ROM που δημιούργησε η διάσημη ομάδα κατασκευής τροποποιημένων Android ROM, ονόματι CyanogenMod[3]. Τα τροποποιημένα ROM από την εν λόγω ομάδα φέρουν βελτιώσεις σε σχέση με το λεγόμενο Pure Android[4] και στοχεύουν στην αύξηση της ταχύτητας, στη μέγιστη δυνατή εξοικονόμηση πόρων και στην πρόσβαση του χρήστη σε όσο το δυνατόν περισσότερες λειτουργίες και ρυθμίσεις του λειτουργικού. Μπορείτε να κατεβάσετε το ROM Eco CM9 από το

http://bit.ly/dh023ecostable

Η επιλογή του συγκεκριμένου ROM βασίστηκε κυρίως στο γεγονός ότι είναι προσαρμοσμένο στις δυνατότητες και στις ιδιαιτερότητες του ZTE Blade. Είναι χτισμένο με γνώμονα την ταχύτητα και τη χαμηλή κατανάλωση ενέργειας, δεδομένου ότι το Blade έχει σχετικά αδύναμο επεξεργαστή. Παράλληλα, η λειτουργικότητα και η απόκριση του κινητού με το συγκεκριμένο ROM είναι άψογες! Ακόμα και το ραδιόφωνο, που με παλαιότερες εκδόσεις του συγκεκριμένου ROM δεν λειτουργούσε, συμπεριφέρεται πλέον τέλεια. Το Eco CM9 βασίζεται στο Android ICS[5] (4.0.4). Η πιο πρόσφατη έκδοση του λειτουργικού, ωστόσο, είναι η Jelly Bean (4.2). Μιλάμε όμως για μια παλιά συσκευή και η "υποχώρηση" στην έκδοση 4.0.4 δεν αποτελεί μειονέκτημα — κάθε άλλο μάλιστα. Για να προσθέσουμε στο λογισμικό Blade και το Google Play, το οποίο δεν ενσωματώνεται στο ECO CM9, θα πρέπει να κατεβάσουμε το Mini Gapps ICS από εδώ:

http://bit.ly/dh023minigapps

Πρώτο βήμα

Για τη συνέχεια θα χρειαστεί να αποκτήσουμε δικαιώματα υπερχρήστη[6] στη συσκευή, όπως επίσης και να εγκαταστήσουμε το Clockwork Recovery. Αν είστε από τα ανήσυχα πνεύματα που έχουν ξεκλειδώσει το σύνολο της RAM, τότε θα έχετε ήδη πραγματοποιήσει αυτά τα βήματα. Καλού κακού, στη συνέχεια θα πούμε δυο λόγια για το πώς επιτυγχάνονται τα παραπάνω…

Κατ' αρχάς, θα χρειαστούμε τα αρχεία recovery-clockwork-2.5.1.8-blade.img, RecoveryManager_v0.29b.apk και UniversalAndroot-1.6.2-beta5.apk. Μπορείτε να τα κατεβάσετε από τα ακόλουθα URLs:

http://bit.ly/dh023recovclockwork

http://bit.ly/dh023recovmanager

---

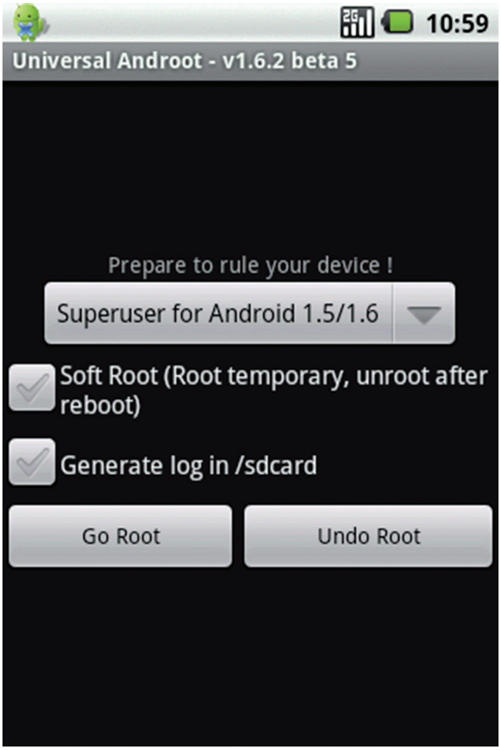
3. Η ομάδα CyanogenMod δεν υποστηρίζει πλέον τη ROM που επιλέξαμε, αλλά αυτό καθόλου δεν μας πτοεί.
4. Έτσι συνηθίζεται να ονομάζεται το "καθαρό" Android, όπως διατίθεται από την ομάδα ανάπτυξης της Google και χωρίς τις παραμετροποιήσεις/προσθήκες τρίτων κατασκευαστών.
5. Το ICS είναι ακρώνυμο του Ice Cream Sandwich. Η Google έχει καθιερώσει να χρησιμοποιεί ονομασίες γλυκών (κατά αλφαβητική σειρά) ως κωδική ονομασία για κάθε έκδοση του λειτουργικού της.
6. Εάν ενδιαφέρεστε να μάθετε περισσότερα σχετικά με τη διαδικασία του rooting, μπορείτε να διαβάσετε το σχετικό άρθρο του Thiseas, στο τεύχος 004 του deltaHacker.

http://bit.ly/dh023uniandroot

Επιπρόσθετα, θα χρειαστούμε κι ένα πρόγραμμα που επιτρέπει την εγκατάσταση εφαρμογών (αρχεία με κατάληξη .apk) απευθείας από την κάρτα μνήμης. Ένα τέτοιο πρόγραμμα είναι το Apk Installer, που μπορούμε να κατεβάσουμε δωρεάν από το Google Play:

https://play.google.com/store/apps/details?id=com.potente.apk.installer

Αφού αντιγράψουμε τα παραπάνω τρία αρχεία στην κάρτα SD του Blade, τρέχουμε το Apk Installer κι εγκαθιστούμε το UniversalAndroot-1.6.2-beta5.apk. Αυτό το πρόγραμμα θα αναλάβει να μας δώσει root access στη συσκευή, με ένα μόλις κλικ! Έτσι, αφού το εγκαταστήσουμε, το τρέχουμε και πατάμε απλά στο Go Root. Ακολούθως, προχωράμε στην εγκατάσταση του RecoveryManager_v0.29b.apk. Η συγκεκριμένη εφαρμογή απαιτεί δικαιώματα root για να εκτελέσει τις λειτουργίες της και γι' αυτό το λόγο θα εμφανιστεί μια σχετική ερώτηση. Εκεί απαντάμε θετικά (προφανώς) κι από το περιβάλλον του προγράμματος επιλέγουμε το Install Recovery. Στη συνέχεια υποδεικνύουμε στην εφαρμογή το αρχείο recovery-clockwork-2.5.1.8-blade.img, που έχουμε τοποθετήσει στην κάρτα μνήμης. Μετά κι από αυτό, είμαστε έτοιμοι να προχωρήσουμε στα επόμενα (και πιο ουσιαστικά) βήματα της διαδικασίας.



Η εφαρμογή androot, εκμεταλλευόμενη μια αδυναμία του συστήματος παρέχει root access πανεύκολα — είτε μονίμως είτε προσωρινά.

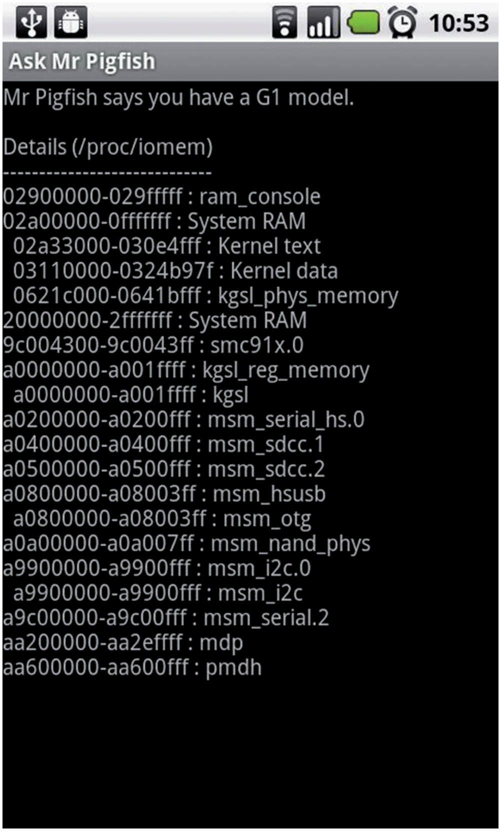
Έλεγχος έκδοσης και προετοιμασία

Σε αυτό το στάδιο πρέπει να αποσαφηνίσουμε αν το Blade που έχουμε στην κατοχή μας είναι τύπου Gen1 ή Gen2. Η διαφορά έγκειται στον τρόπο που είναι χωρισμένα τα διαμερίσματα στην εσωτερική μνήμη του κινητού. Η διευθέτηση των διαμερισμάτων (των κατατμήσεων, δηλαδή) παίζει σημαντικό ρόλο, καθώς θα χρειαστεί να τροποποιήσουμε κατάλληλα τα μεγέθη τους ώστε να χωρέσει το νέο ROM. Για να ελέγξουμε τον τύπο της συσκευής θα χρησιμοποιήσουμε το πρόγραμμα Ask Mr Pigfish, το οποίο μπορούμε να πάρουμε δωρεάν από το Google Play

https://play.google.com/store/apps/details?id=com.apedroid.pigfish

Μόλις το εγκαταστήσουμε και το εκτελέσουμε, θα μάς δώσει αμέσως την πληροφορία που ζητάμε.

Πριν προχωρήσουμε στο partitioning του αποθηκευτικού χώρου του Blade, αξίζει να δούμε ποια διαμερίσματα υπάρχουν ήδη στην εσωτερική μνήμη και το ρόλο του καθενός.



Το λιτό περιβάλλον του Ask Mr Pigfish μάς πληροφορεί άμεσα αν το Blade μας είναι Gen1 ή Gen2.

- **Boot partition.** Εδώ φιλοξενούνται τα απαραίτητα αρχεία για την εκκίνηση του συστήματος, καθώς και οι πληροφορίες για τα υπόλοιπα partitions αλλά και την ίδια τη συσκευή.
- **System partition.** Εδώ κατοικεί το υπόλοιπο λειτουργικό σύστημα, οι βασικές του εφαρμογές, οι βιβλιοθήκες, οι ρυθμίσεις, τα αρθρώματα του πυρήνα και το τυχόν προεγκατεστημένο λογισμικό.
- **Recovery partition.** Το Android είναι ευέλικτο και φιλικό ως προς τον developer. Έτσι, στην περίπτωση που υπάρξει κάποιο πρόβλημα με τη φόρτωση του λειτουργικού από το System partition, το σύστημα δοκιμάζει να ξεκινήσει από αυτό το διαμέρισμα. Μπορούμε να επιβάλλουμε την εκκίνηση από το Recovery partition, όταν κατά την ενεργοποίηση της συσκευής κρατάμε πατημένο το κουμπί μείωσης της έντασης του ήχου [-].
- **Cache partition.** Εδώ αποθηκεύονται τα δεδομένα της προσωρινής μνήμης.
- **Data partition.** Το Data partition είναι ο χώρος που φιλοξενεί τα δεδομένα του χρήστη, καθώς και τις εφαρμογές που ο ίδιος εγκαθιστά.

Το ROM που έχουμε επιλέξει προορίζεται για συσκευές τύπου Gen2. Έτσι, αν η συσκευή μας είναι τύπου Gen1, θα πρέπει να τροποποιήσουμε κατάλληλα το layout των κατατμήσεων. Σε αυτή τη διαδικασία θα μας βοηθήσει το πρόγραμμα TPT Helper, το οποίο επίσης μπορούμε να κατεβάσουμε από το Google Play:

`https://play.google.com/store/apps/details?id=com.amphoras.tpthelper`

Σημειώστε ότι η εργασία του εν λόγω προγράμματος προβλέπει τη λήψη ορισμένων αρχείων από το δίκτυο. Επομένως, καθ' όλη τη διάρκεια της σχετικής διαδικασίας, πρέπει να έχουμε το WiFi ενεργοποιημένο. Αφού εγκαταστήσουμε το TPT helper, το τρέχουμε και φροντίζουμε ώστε ν' απαντήσουμε σωστά στις ερωτήσεις που θα μας απευθύνει. Αμέσως μετά πρέπει να δηλώσουμε τα επιθυμητά μεγέθη για τις κατατμήσεις του συστήματος. Αν η συσκευή μας είναι Gen2, πρέπει να επιλέξουμε τα εξής: 2MB για το Cache partition, 160MB για το System partition, 294MB για το Data partition και 0MB για το λεγόμενο OEM partition. Αν η συσκευή μας είναι Gen1, αρκεί να επιλέξουμε το Gen1 to Gen2. Όπως αναφέραμε, το πρόγραμμα θα αρχίσει να κατεβάζει τα απαιτούμενα αρχεία από το δίκτυο και θα ζητήσει δικαιώματα root για να προχωρήσει. Τώρα πρέπει να απενεργοποιήσουμε και να ενεργοποιήσουμε εκ νέου τη συσκευή, κρατώντας πατημένο το πλήκτρο αύξησης της έντασης του ήχου

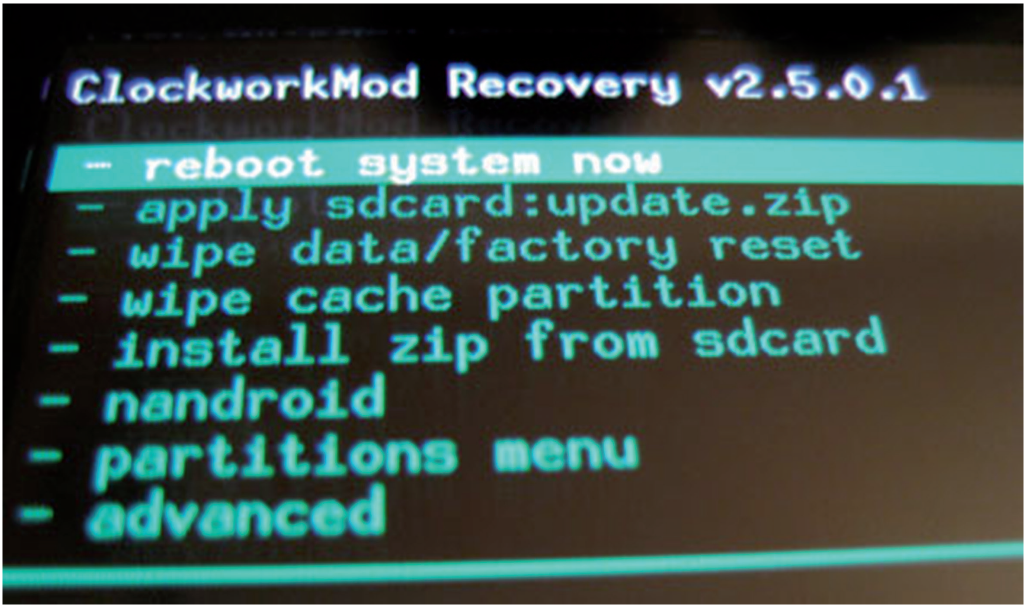
[+] και το πλήκτρο Menu. Κατά την εκκίνηση της συσκευής θα δούμε τη διαδικασία του repartitioning.

Εικόνα: tpt.jpg

Το TPT Helper αναλαμβάνει να κάνει όλη τη "βρόμικη" αλλά απαραίτητη δουλειά της αλλαγής του layout και του repartitioning, ώστε να μπορέσουμε να εγκαταστήσουμε το custom ROM.

Εγκατάσταση ROM

Όταν ξεκινήσει το σύστημα μπαίνουμε στο περιβάλλον του ClockWork. Η μετακίνηση στα στοιχεία του μενού του ClockWork γίνεται με τα πλήκτρα ρύθμισης της έντασης του ήχου, ενώ η επιβεβαίωση/επιλογή πραγματοποιείται με το κουμπί Home. Μεταβαίνουμε στην επιλογή Setup from zip κι αφού εντοπίσουμε στην κάρτα μνήμης το αρχείο ZIP με το νέο ROM (Eco-CM9-20130511-stable-blade.zip), επιλέγουμε την εγκατάστασή του. Μετά από λίγα λεπτά, δάκρυα συγκίνησης θα αρχίσουν να κατακλύζουν το πρόσωπό μας: Το παλιό, καλό μας Blade, θα τρέχει πλέον το Android ICS (4.0.4)!



Το menu του Clockwork Recovery, πέρα από την επαναφορά του συστήματος επιτρέπει την εγκατάσταση νέου ROM ή ενημερώσεων, που βρίσκονται στην κάρτα μνήμης.

Πριν μας συνεπάρει η χαρά για το καινούργιο λειτουργικό, πρέπει να κάνουμε ένα ακόμα βήμα και στο μενού της συσκευής μας να εγκαταστήσουμε το Google Play. Για το σκοπό αυτό κλείνουμε και ξανανοίγουμε το κινητό, κρατώντας πατημένο το πλήκτρο μείωσης της έντασης [-]. Εναλλακτικά, μπορούμε να πατήσουμε το κουμπί On/Off στο μενού του CM9 κι από εκεί να επιλέξουμε την επανεκκίνηση σε Recovery Mode! Αφού επανεκκινήσουμε το κινητό σε recovery mode, επισκεπτόμαστε πάλι το Setup from zip κι αναζητούμε, στην κάρτα SD, το αρχείο με την εφαρμογή του Google Play (mini-gapps-ics-20120429-signed.zip). Τώρα, αρκεί να πραγματοποι-



Οι πληροφορίες συστήματος για το ανανεωμένο Blade!

ήσουμε τη σχετική εγκατάσταση και να επανεκκινήσουμε το κινητό μας. Αυτό ήταν! Μπορούμε πλέον να απολαύσουμε το ανανεωμένο μας Blade.

Πέρα από τον αέρα ανανέωσης, τη βελτίωση στην ταχύτητα και τις νέες εφαρμογές που μπορούμε να προσθέσουμε στη συσκευή, μεγάλο ενδιαφέρον παρουσιάζουν και οι διαθέσιμες ρυθμίσεις για το σύστημα. Μεταξύ άλλων μπορούμε να καθορίσουμε πλήρως τον τρόπο διαχείρισης της RAM, όπως και να υπερχρονίσουμε τον επεξεργαστή της συσκευής! Αυτή η επιλογή βέβαια χρειάζεται σύνεση: Όσο αυξάνουμε τη συχνότητα του επεξεργαστή τόσο αυξάνεται και η κατανάλωση ενέργειας, με συνέπεια τη μείωση της αυτονομίας της συσκευής. Εξάλλου, το υπερβολικό "πουσάρισμα" του επεξεργαστή καθιστά τη συσκευή ασταθή. Εμείς πάντως σας προτρέπουμε να πειραματιστείτε ελεύθερα και να φέρετε το κινητό στα μέτρα σας. Τελικά το γερο-Blade δεν τα πάει κι άσχημα για low-budget συσκευή τριετίας... Δεν συμφωνείτε;



Skill: Intermediate
Tags: Windows, OpenBSD, PF, NAT, DHCP, port forwarding, DNS, loopback, virtualization, VirtualBox, bridging, Tor, anonymity, dnsmasq

by sub2raw



WinBSD: Ένας γάμος που θέλετε να γίνει!

Τα Windows τα ξέρετε πολύ καλά. Το OpenBSD πιθανώς να το γνωρίζετε κι αυτό. Αν όχι, ίσως να έχετε πάρει μια ιδέα περί τίνος πρόκειται παρακολουθώντας το deltaCast s01e10. Σε κάθε περίπτωση μιλάμε για δύο εντελώς διαφορετικά λειτουργικά συστήματα, τόσο από πλευράς αρχιτεκτονικής όσο κι από πλευράς χρηστικής φιλοσοφίας. Κι όμως. Μπορούν να συνεργαστούν αρμονικά μεταξύ τους — και μάλιστα κατά τρόπο που μάλλον δεν φαντάζεστε.

Γράψαμε «μάλλον δεν φαντάζεστε», γιατί υποθέσαμε πως θα σκεφτήκατε κάτι στο στιλ «ναι, ο-κέι, στήνουμε ένα μποξ με όπεν μπι-ες-ντι και το κάνουμε ρούτερ και φάιργουολ για ένα δίκτυο με γουίντοουζ πι-σιζ — σιγά την πρωτότυπη ιδέα». Κοιτάξτε, επειδή δεν είμαστε πάντα καλοί στο να φανταζόμαστε τι φανταζόσαστε, ξεχάστε ό,τι διαβάσατε μόλις. Τώρα που το σκεφτόμαστε καλύτερα, είμαστε βέβαιοι ότι έχετε καταλάβει που το πάμε: Φυσικά και *δεν* συζητάμε για το απλό στήσιμο ενός OpenBSD-based router/firewall. Αντίθετα, σκοπεύουμε να ξηλώσουμε το όποιο application firewall χρησιμοποιούν τα Windows και στη θέση του να βάλουμε το PF, δηλαδή το πανίσχυρο firewall του OpenBSD. Βέβαια το PF δεν μπορεί να σταθεί από μόνο του κάτω από τα Windows, οπότε η ανακαίνιση του συστήματος θα είναι περισσότερο εκ βάθρων. Συγκεκριμένα, σε πολύ λίγο θα δείξουμε πώς παραμερίζουμε το firewall των Windows και στη θέση του θρονιάζουμε μια ολόκληρη εγκατάσταση του OpenBSD, με το PF ρυθμισμένο όπως θέλουμε καθώς κι όποια άλλα services επιθυμούμε — πάντα στην υπηρεσία του Windows PC. Περιττό να πούμε ότι τα Windows και οι εφαρμογές τους *δεν* θα βγαίνουν στον έξω κόσμο (τοπικό δίκτυο, Internet) απευθείας, αλλά *μέσω* του OpenBSD. Και κάπου εδώ, φίλες και φίλοι, το ακόλουθο ερώτημα προκύπτει κατά τρόπο φυσιολογικό...

Γιατί πάνω στη Γη θα ήθελα να κάνω κάτι σαν αυτό;[1]

Θα σας αναφέρουμε μερικούς από τους λόγους που μπορούμε να σκεφτούμε, όχι κατ’ ανάγκη με σειρά σπουδαιότητας.

- Το application firewall που έχουμε στα Windows, είτε πρόκειται για το ενσωματωμένο είτε για εκείνο κάποιου τρίτου προμηθευτή, πιθανότατα είναι μια χαρά. Το PF, από την άλλη, είναι αυτό που λέμε industrial strength. Κάποιοι routers που βασίζονται σ’ αυτό κοστίζουν εκατοντάδες ή ακόμη και χιλιάδες ευρώ, εμείς όμως μπορούμε να έχουμε το PF δωρεάν, αφού *είναι* δωρεάν.
- Από πλευράς ασφαλείας, πιθανώς να υπάρχουν περισσότερα zero-days για το firewall που χρησιμοποιείτε στα Windows σας, παρά για το PF του OpenBSD. Προσέξτε: 100% σίγουροι για την υπόθεσή μας δεν μπορούμε να είμαστε, φλερτάρουμε όμως άνετα με το 99,37%.
- Ο συνδυασμός OpenBSD+PF δεν καταναλώνει πολλούς πόρους από το σύστημα. Για παράδειγμα, τα 32MB μνήμης RAM είναι αρκετά. Αλήθεια, το application firewall που χρησιμοποιείτε πόση μνήμη πιάνει;
- Ο καλύτερος λόγος για να βάλετε OpenBSD στα Windows σας, είναι επειδή μπορείτε. Για να διαβάζετε αυτό το περιοδικό καταλαβαίνετε τι εννοούμε, οπότε δεν χρειάζεται να πούμε τίποτε περισσότερο επ’ αυτού.

Ναι μεν, αλλά...

Ωραία ακούγονται τα προηγούμενα, όμως το application firewall στα Windows μπορεί να κάνει κάτι που το PF αδυνατεί: Είναι σε θέση να επιτρέπει ή όχι την πρόσβαση στο δίκτυο ανά εφαρμογή. Το PF, από την άλλη, δεν ξέρει τίποτε από εφαρμογές. Το μόνο που μπορεί να κάνει —και μάλιστα άριστα— είναι να φιλτράρει δικτυακά πακέτα. Εξ ου και τ’ όνομά του: *Packet Filter*. Ενώ λοιπόν το PF μπορεί να παίρνει αποφάσεις του στιλ «όλα τα πακέτα που βγαίνουν στον έξω κόσμο άφηνέ τα», «μπλόκαρε τα εξερχόμενα πακέτα που έρχονται από τη τάδε διεύθυνση IP», «μπλόκαρε όλα τα

1. From the why-on-Earth-would-I-want-to-do-something-like-that department.

ξεκάρφωτα εισερχόμενα πακέτα εκτός από εκείνα που φτάνουν στο port 22» κ.ο.κ., δεν μπορεί να παίρνει αποφάσεις του στιλ «εμπόδισε το τάδε εκτελέσιμο να βγει στο Internet».

Πράγματι, έτσι είναι τα πράγματα και δεν μπορεί να γίνει τίποτε γι' αυτό. Τώρα, οι χρήστες των Windows που *δεν* νοιάζονται για την επιλεκτική πρόσβαση στον έξω κόσμο σε επίπεδο εφαρμογής, λίγο θα νοιαστούν. Πιστεύουμε μάλιστα ότι δεν διαβάζουν καν αυτές τις γραμμές και βρίσκονται ήδη στην επόμενη ενότητα.

Όλοι οι υπόλοιποι, όμως, τι κάνουν; Εγκαταλείπουν την ιδέα του συνδυασμού OpenBSD+PF για το Windows PC; Όχι απαραίτητα. Βλέπετε, κάτι που δεν αναφέραμε προηγουμένως είναι ότι κάλλιστα μπορούμε να έχουμε το application firewall ενεργοποιημένο και *ταυτόχρονα* το OpenBSD με το PF εγκατεστημένα! Απλά, σε μια τέτοια περίπτωση το PF θ' αποτελεί ένα έξτρα τείχος προστασίας, μπροστά από το όποιο application firewall των Windows. Αυτό δεν είναι κατ' ανάγκη κακό. Θυμηθείτε την προηγούμενη υπόθεσή μας: Κατά πάσα πιθανότητα, τα zero-days για το όποιο application firewall είναι περισσότερα από 'κείνα για το PF.

Δεν υπάρχουν δικαιολογίες, λοιπόν: Όσοι τολμηροί, σηκώστε τα μανίκια κι ακολουθήστε μας!

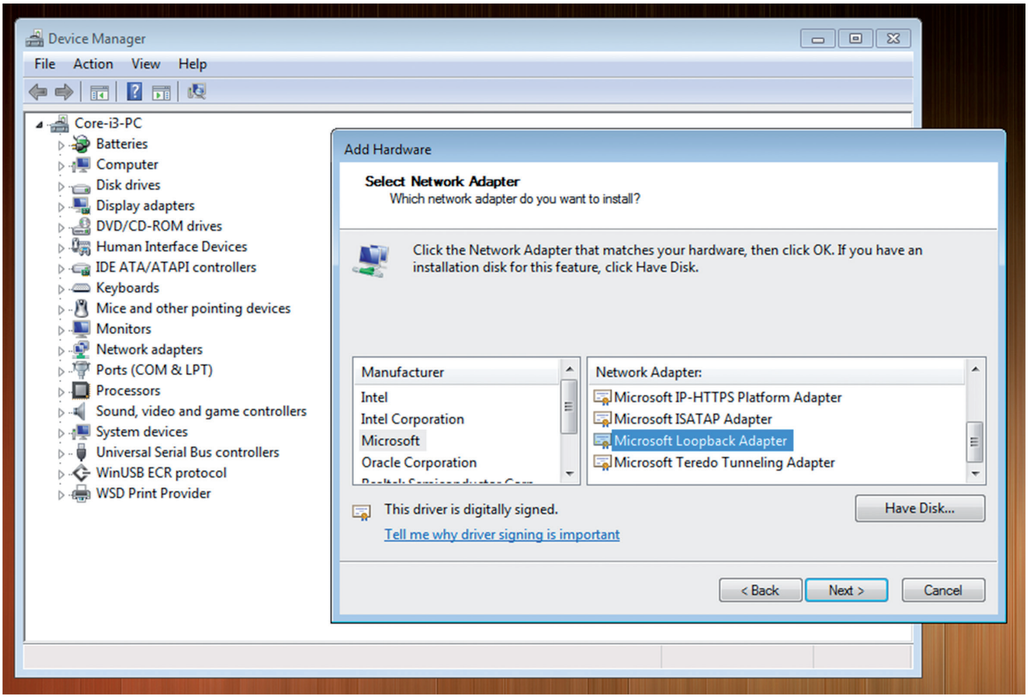
Οι προθέσεις, το σχέδιο και το περιβάλλον

Προκειμένου να περιγράψουμε με ακρίβεια το αποτέλεσμα που επιθυμούμε να πετύχουμε, αναφέρουμε γενικά και σε σημεία τις ενέργειες που θα κάνουμε σε λίγο.

- Εγκατάσταση του OpenBSD σε VirtualBox VM, στο PC με τα Windows φυσικά.
- Ρύθμιση του guest OS, δηλαδή του OpenBSD, ώστε να λειτουργεί ως NAT router.
- Ρύθμιση κι ενεργοποίηση DHCP server στο OpenBSD, ώστε να μοιράζει αυτόματα διευθύνσεις IP στους clients. Κάτι τέτοιο δεν είναι υποχρεωτικό, αφού ο βασικός πελάτης του OpenBSD θα είναι το host OS, δηλαδή τα Windows. Αν όμως θέλουμε να σηκώνουμε κι άλλα VMs, τα οποία θα βγαίνουν στον έξω κόσμο μέσω του OpenBSD, τότε η παρουσία DHCP server είναι εξαιρετικά βολική.
- Εγκατάσταση του Microsoft Loopback Adapter στο host OS.
- Ρύθμιση της δικτύωσης του OpenBSD, ώστε ο WAN adapter να είναι bridged με τον φυσικό network adapter του PC, τον οποίο μέχρι στιγμής χρησιμοποιούν τα Windows για να βγαίνουν στο δίκτυο. Επίσης: Ρύθμιση του LAN adapter του OpenBSD, ώστε να είναι bridged με τον Microsoft Loopback Adapter.
- Απενεργοποίηση του TCP/IP για τον υπάρχοντα network adapter του PC, αυτόν δηλαδή που ως τώρα χρησιμοποιούσαν τα Windows για να βγαίνουν στο Internet.
- Για τα Windows, ο μόνος ενεργός network adapter που υποστηρίζει TCP/IP είναι πλέον ο Loopback κι αυτός είναι bridged με τον LAN adapter του OpenBSD VM. Κατά συνέπεια, τα Windows βγαίνουν στον έξω κόσμο μέσω του OpenBSD-based NAT router (που έχει και DHCP server).
- Ρύθμιση του PF στο OpenBSD VM, ώστε να προστατεύονται τα Windows.
- Προαιρετικά: Απενεργοποίηση του όποιου application firewall χρησιμοποιούσα-

με ως τώρα στα Windows.

- Μέριμνα ώστε το OpenBSD VM να συμπεριφέρεται ως Windows service και να ξεκινά αυτόματα, κατά την εκκίνηση του host OS. Προφανώς, θα απενεργοποιείται αυτόματα και κατά το shutdown — πάντα του host OS.
- Προαιρετικά: Μέριμνα ώστε τα Windows να παίρνουν πάντα το ίδιο IP. Στη συνέχεια: Κανόνες port forwarding στο WAN interface του OpenBSD VM, ώστε άλλα μηχανήματα του τοπικού δικύου να έχουν πρόσβαση σε αρχεία κι εκτυπωτές που ενδεχομένως μοιράζονται από το Windows box.
- Προαιρετικά: Στατικό IP για το WAN interface του OpenBSD VM, ώστε, π.χ., να είναι εφικτός ο ορισμός σχετικών κανόνων port forwarding στον αληθινό router, μπροστά από το Windows box.
- Προαιρετικά: Μετατροπή του OpenBSD VM σε Tor middlebox, ώστε το host OS ή/και κάποια από τα VirtualBox VMs που βγαίνουν στο Internet μέσω του OpenBSD να χρησιμοποιούν αυτομάτως το Tor network.
- Προαιρετικά: Εγκατάσταση στο OpenBSD VM ενός nameserver, ο οποίος θα εξυπηρετεί ένα ή περισσότερα μηχανήματα πίσω του. Υπογραμμίζουμε ότι ο nameserver *δεν* θα εξυπηρετεί μηχανήματα που χρησιμοποιούν το Tor network, αφού τα DNS queries αυτών θέλουμε επίσης να περνάνε μέσα από το δίκτυο ανωνυμίας του Tor.



Η εγκατάσταση του Microsoft Loopback Adapter ξεκινά από τον Device Manager, με δεξί κλικ πάνω στο όνομα του υπολογιστή (για εμάς Core-i3-PC) κι επιλογή του Add legacy hardware. Η παρουσία του Loopback Adapter είναι απαραίτητη, ώστε μετά τις κατάλληλες ρυθμίσεις τα Windows να μη βγαίνουν στο δίκτυο απευθείας αλλά μέσω του OpenBSD VM.

Τις δικές μας δοκιμές τις κάναμε σε ένα PC με τα 64μπιτα Windows 7 Premium, στο οποίο ως application firewall χρησιμοποιούσαμε το Windows Firewall της Microsoft. Η κεντρική μνήμη RAM του PC είχε μέγεθος 8GB. Το μηχάνημα διέθετε μία μόνο κάρτα δικτύου Ethernet και συνεπώς έβγαινε στο δίκτυο ενσύρματα, μέσω του VDSL modem/router που υπήρχε στο LAN. Επιλέξαμε την πλέον πρόσφατη έκδοση του VirtualBox (4.2.16) και σε VM εγκαταστήσαμε την 32μπιτη εκδοχή του OpenBSD 5.3.

Πριν περάσουμε στη δράση, να σημειώσουμε ότι γι' αυτό που θέλουμε να πετύχουμε το OpenBSD δεν αποτελεί μονόδρομο. Στη θέση του μπορούμε να έχουμε ένα οποιοδήποτε λειτουργικό σύστημα που είναι κατάλληλο για routing — και το pfSense είναι το πρώτο που μας έρχεται στο μυαλό (βλ. και `http://deltahacker.gr/deltacast-s01e07`). Ένα από τα καλά του pfSense είναι η μεγάλη ευκολία χειρισμού που προσφέρει. Επιπλέον, στο pfSense έχουμε τη δυνατότητα να προσθέτουμε —επίσης πανεύκολα— έξτρα λειτουργίες και δυνατότητες. Μπορούμε, π.χ., να του εγκαταστήσουμε το Intrusion Detection System (IDS) ονόματι Snort. Σε σύγκριση με το OpenBSD, όμως, το pfSense είναι αρκετά πιο βαρύ. Αν, για παράδειγμα, θέλουμε το Snort, τότε το VM του pfSense προτείνεται να 'χει τουλάχιστον 1GB RAM. Αλλά και η βασική εγκατάσταση του pfSense καλό είναι να γίνεται σε VM/box με τουλάχιστον 128MB RAM. Σίγουρα δεν πρόκειται για απαγορευτική ποσότητα, μόνο που, και πάλι, είναι τέσσερεις φορές περισσότερη από την ποσότητα μνήμης που απαιτεί το OpenBSD.

Στο υπόλοιπο του άρθρου δείχνουμε, αναλυτικά και βήμα προς βήμα, όλες τις ενέργειες που κάναμε προκειμένου ν' αντικαταστήσουμε το Windows Firewall της Microsoft με τον συνδυασμό OpenBSD+PF.

Βήμα 0 — Δημιουργία κατάλληλου VirtualBox VM

Το OpenBSD είναι αρκετά ελαφρύ κι επομένως η λειτουργία του υπό τα Windows έχει ελάχιστο αντίκτυπο στο όλο σύστημα. Ακολουθούν οι προδιαγραφές του VirtualBox VM που φτιάξαμε, προκειμένου να εγκαταστήσουμε εκεί το OpenBSD:

```
=================================================================
Type:                      BSD
Version:                   OpenBSD
Base Memory:               32MB
Hardware clock:            UTC time
Processors:                1
Video Memory:              9MB
Storage:                   32GB, VDI, dynamically allocated
Audio:                     Disabled
Network Adapter 1:         Bridged Adapter
Network Adapter 2:         Host-only Adapter
USB Controller:            Disabled
=================================================================
```

Θα παρατηρήσατε ότι, σε αντίθεση με όσα έχουμε ήδη συζητήσει, ο Network Adapter 2 του VM, ο οποίος θα 'χει ρόλο LAN adapter για τον router μας, είναι τύπου

host-only κι όχι bridged. Πράγματι, προς το παρόν θέλουμε να εγκαταστήσουμε το OpenBSD, να το ρυθμίσουμε ώστε να δουλεύει ως NAT router και φυσικά να κάνουμε τις δοκιμές μας από άλλα VMs. Γι' αυτές τις δουλειές ο LAN adapter αρκεί να 'ναι τύπου host-only. Αργότερα, που θα έχουμε εγκαταστήσει και τον Microsoft Loopback Adapter στα Windows, θα αλλάξουμε τον τύπο του Network Adapter 2 σε bridged.

Βήμα 1
Εγκατάσταση του OpenBSD

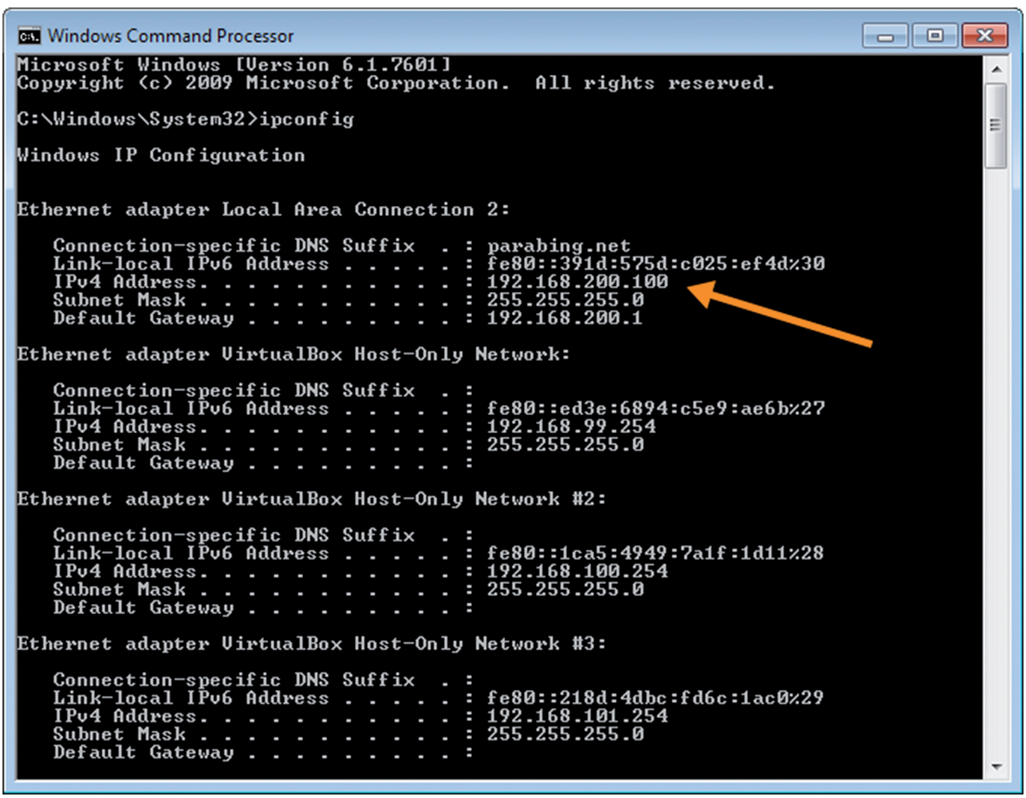
Η διαδικασία της εγκατάστασης του OpenBSD επιδεικνύεται αναλυτικά στο 10ο επεισόδιο της 1ης σεζόν του deltaCast, το οποίο μπορείτε να παρακολουθήσετε στο

http://deltahacker.gr/deltacast-s01e10

Βήμα 2
Μετατροπή του OpenBSD σε NAT router με DHCP

Μετά την εγκατάσταση του OpenBSD και μια πρώτη γνωριμία με το νέο περιβάλλον, στο deltaCast s01e10 δείχνουμε πώς μετατρέπουμε το λειτουργικό σε NAT router με υποστήριξη DHCP. Παρακαλούμε, εφαρμόστε τις οδηγίες του επεισοδίου.

Τα Windows έχουν πάρει την πρώτη διαθέσιμη από τις διευθύνσεις IP που μοιράζει ο DHCP server, στο OpenBSD VM. Παρατηρήστε επίσης τη διεύθυνση του λεγόμενου Default Gateway. Πλέον, το host OS, δηλαδή τα Windows μας, δεν βγαίνουν απευθείας στο δίκτυο!



Βήμα 3
Νέο ruleset για το PF

Το OpenBSD που μόλις εγκαταστήσαμε και ρυθμίσαμε σε VM, το θέλουμε φυσικά σε ρόλο firewall για το Windows PC μας. Στο deltaCast s01e10 δείχνουμε πώς φέρνουμε το PF στα μέτρα μας, ώστε να παρέχει άριστη προστασία στην ίδια την εγκατάσταση του OpenBSD, αλλά και σε όλες τις άλλες συσκευές που βγαίνουν στον έξω κόσμο μέσω αυτής. Για ευκολία, παραθέτουμε εδώ τα περιεχόμενα του αρχείου /etc/pf.conf:

```
wan_if="em0"
lan_if="em1"

allowed_icmp_types="echoreq"
allowed_tcp_service_ports="{ 22 }"

set block-policy drop
set loginterface $wan_if
set skip on lo

match out on $wan_if inet from !($wan_if:network) to any nat-to ($wan_if:0)

block in log
pass out quick

antispoof quick for { lo $lan_if }

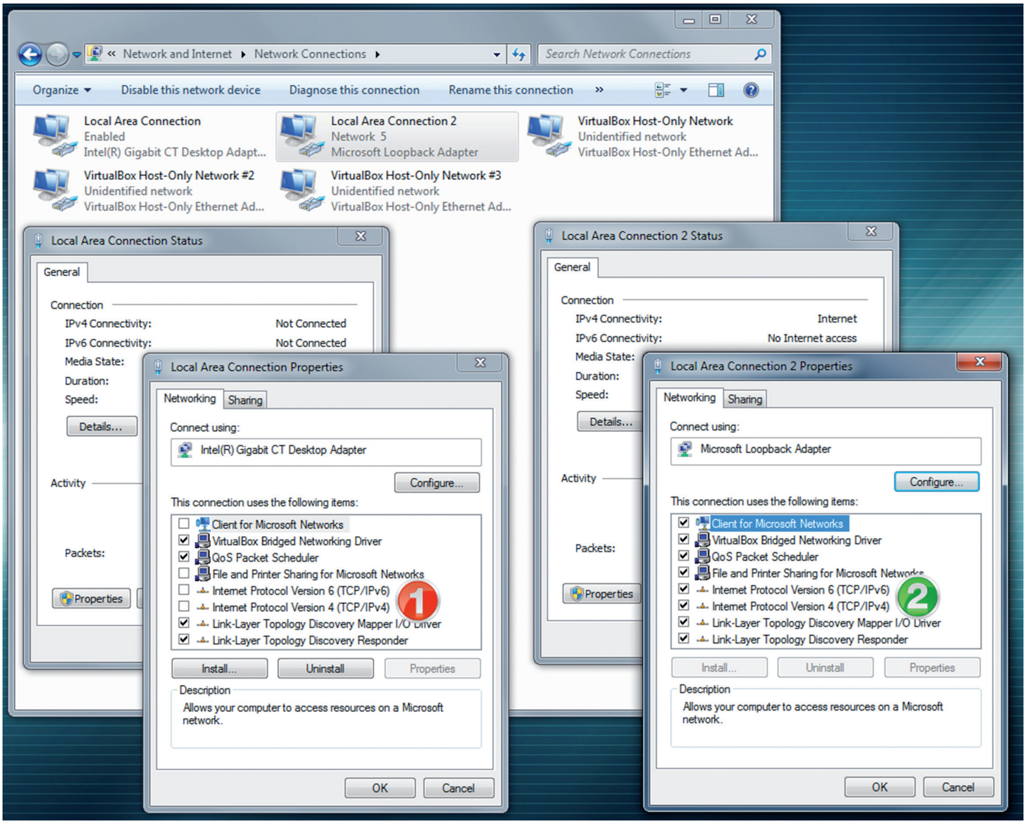
pass in on $wan_if inet proto tcp from any \
    to ($wan_if) port $allowed_tcp_service_ports

pass in inet proto icmp all icmp-type $allowed_icmp_types

pass in on $lan_if
```

Ίσως είναι λίγο διαφορετικά από τα περιεχόμενα του /etc/pf.conf που δείχνουμε στο deltaCast —κάποιοι έχουν μια τάση να πειράζουν σχεδόν τα πάντα, σχεδόν όλη την ώρα—, ωστόσο οι βασικές κατευθύνσεις παραμένουν ίδιες:

- τα πακέτα που ξεκινούν από το LAN δεν εμποδίζονται, όπου κι αν κατεθύνονται
- τα ξεκάρφωτα πακέτα που φτάνουν στο WAN interface από τον έξω κόσμο απορρίπτονται και μάλιστα χωρίς να ειδοποιείται ο αποστολέας, εκτός κι αν προορίζονται για το port 22, που εξ ορισμού χρησιμοποιεί η υπηρεσία του OpenSSH, οπότε σ’ αυτή την περίπτωση περνάνε
- για τα πακέτα που ξεκινάνε από το εσωτερικό δίκτυο κι έχουν προορισμούς εκτός LAN, λαμβάνει χώρα το λεγόμενο Network Address Translation (NAT)
- τα πακέτα ICMP τύπου echo request επιτρέπονται σε κάθε network interface



Τη σύνδεση με όνομα Local Area Connection, που αφορά στην κάρτα Ethernet του (φυσικού) υπολογιστή, χρησιμοποιούσαν μέχρι πρότινος τα Windows για να βγαίνουν στον έξω κόσμο. Απενεργοποιώντας, όμως, τα TCP/IPv4 και TCP/IPv6 για το Local Area Connection (1), τα Windows στρέφονται στη σύνδεση με όνομα Local Area Connection 2, η οποία αφορά στον Microsoft Loopback Adapter κι έχει ενεργοποιημένα τα TCP/IPv4 και TCP/IPv6 (2).

Βήμα 4
Δοκιμές από άλλα VMs

Με το OpenBSD VM ενεργοποιημένο, ξεκινάμε ένα οποιοδήποτε άλλο VM που έχει έναν μόνο network adapter τύπου host-only[2] και το guest OS του έτσι ρυθμισμένο, ώστε να παίρνει αυτόματα IP (μέσω DHCP). Αν όλα έχουν πάει καλά με το OpenBSD VM, τότε από το άλλο VM θα πρέπει να είμαστε σε θέση...

- να κάνουμε ping στα LAN και WAN interfaces του OpenBSD και να παίρνουμε απαντήσεις
- να κάνουμε ping σε servers στο Internet και να παίρνουμε απαντήσεις
- να σερφάρουμε όπου θέλουμε χωρίς προβλήματα και γενικά να χρησιμοποιούμε οποιαδήποτε ιντερνετική υπηρεσία, επίσης χωρίς προβλήματα
- να συνδεόμαστε, μέσω SSH, στο port 22 του OpenBSD LAN adapter

2. Για την ακρίβεια, με έναν adapter στο ίδιο host-only network που βρίσκεται κι ο LAN adapter του OpenBSD VM.

Επιπρόσθετα, από τα Windows ή οποιονδήποτε άλλον υπολογιστή του (αληθινού) φυσικού δικτύου θα μπορούμε να συνδεόμαστε, επίσης μέσω SSH, στο port 22 του OpenBSD WAN interface, ενώ ένα port scan στη διεύθυνση IP του προαναφερθέντος θα δείχνει μόνο το port 22 ανοικτό.

Αν κάτι από τα προηγούμενα δεν δουλεύει, τότε το πρόβλημα πιθανώς θα εντοπίζεται στους κανόνες του PF. Σε κάθε περίπτωση, πριν προχωρήσουμε βεβαιωθείτε ότι όλα είναι όμορφα κι ωραία. Το πολύ πολύ να παρακολουθήσετε για άλλη μια φορά το deltaCast s01e10. Είναι οδυνηρή η εμπειρία, ξέρουμε, ωστόσο μερικές φορές είναι κι απαραίτητη.

Βήμα 5
Εγκατάσταση Microsoft Loopback Adapter

Στον Windows host ανοίγουμε το Control Panel κι ακολουθούμε τη διαδρομή System and Security → System → Device Manager (βλ. πάνω αριστερά). Στο νέο παράθυρο που θα εμφανιστεί, στην κορυφή του δέντρου των συσκευών κάνουμε δεξί κλικ πάνω στο όνομα του υπολογιστή, επιλέγουμε το Add legacy hardware κι αμέσως εμφανίζεται ο Add Hardware Wizard. Ξεκινάμε με κλικ στο Next κι επιλέγουμε το Install the hardware that I manually select from a list (Advanced). Άλλο ένα κλικ στο Next και, παρακάτω, στο πλαίσιο Common hardware types, επιλέγουμε το Network adapters. Στην επόμενη οθόνη θα δούμε δύο πλαίσια. Σ' εκείνο που είναι αριστερά (Manufacturer) κάνουμε κλικ πάνω στο Microsoft. Από το πλαίσιο δεξιά επιλέγουμε το Microsoft Loopback Adapter και μετά κάνουμε ξανά κλικ στο Next. Στην επόμενη οθόνη θα έχουμε μια τελευταία ευκαιρία να δούμε τι πρόκειται να εγκατασταθεί (ο Microsoft Loopback Adapter). Μ' ένα τελευταίο κλικ στο Next η διαδικασία ξεκινά κι ολοκληρώνεται σε λίγα δευτερόλεπτα. Στην τελευταία οθόνη του wizard θα πληροφορηθούμε για το τι έγινε (τι εγκαταστάθηκε) και θα πατήσουμε στο Finish.

Βήμα 6
Bridging του OpenBSD LAN adapter με τον Loopback

Πριν γίνει το bridging μεταξύ Microsoft Loopback Adapter και Network Adapter 2 του VM (LAN adapter), φροντίζουμε ώστε ο τελευταίος να έχει IP που δεν εμπίπτει σε κανένα από τα host-only networks που υφίστανται στο host OS, δηλαδή στα Windows. Εμείς, π.χ., στη δική μας εγκατάσταση του VirtulBox είχαμε ορίσει τρία διαφορετικά host-only networks: 192.168.99.0/24, 192.168.100.0/24 και 192.168.101.0/24. Ακριβώς γι' αυτό το λόγο, φροντίσαμε ώστε

α) το IP του LAN adapter να είναι το 192.168.200.1,

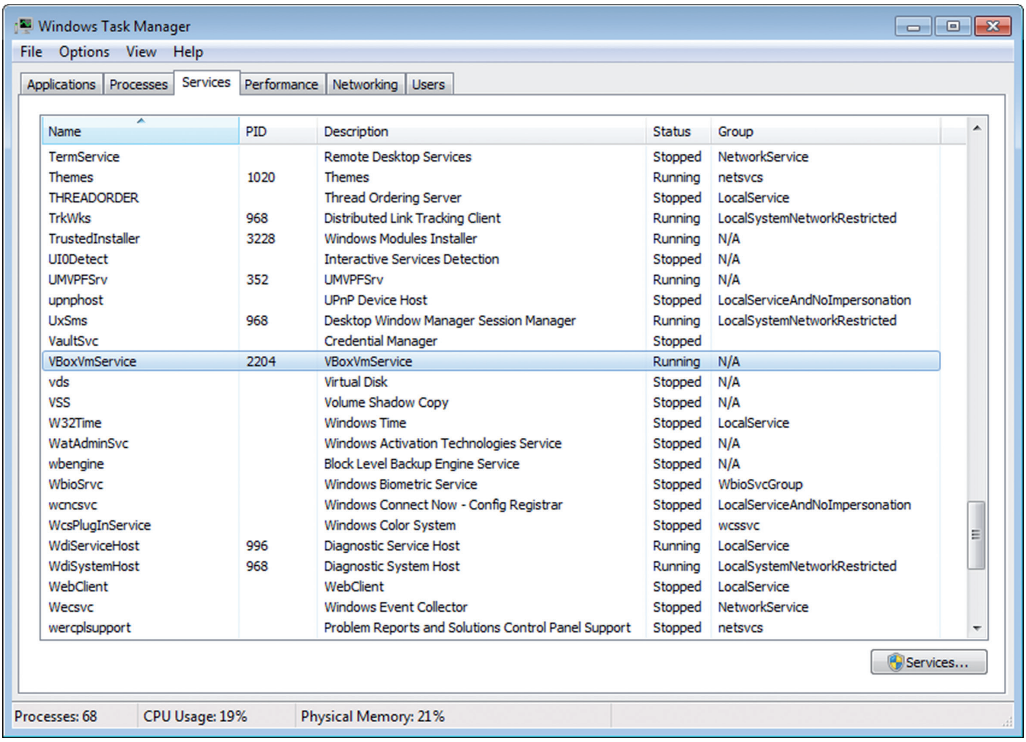
β) ο DHCP server του OpenBSD να μοιράζει διευθύνσεις IP από το 192.168.200.100 έως το 192.168.200.150.

Δείτε πώς γίνεται μια τέτοια αλλαγή. Αρχικά, από την κονσόλα του OpenBSD VM κάνουμε login στον λογαριασμό του root. Για τη συγκεκριμένη εγκατάσταση, το LAN interface είναι το em1. Ανοίγουμε λοιπόν με έναν text editor το αρχείο /etc/hostname.em1 και φροντίζουμε ώστε να έχει τα ακόλουθα περιεχόμενα:

```
inet 192.168.200.1 255.255.255.0
```

Ακολούθως, τροποποιούμε κατάλληλα και το αρχείο /etc/hosts. Παρατηρήστε τη σχετική γραμμή από το δικό μας:

```
192.168.200.1          therouter.parabing.net        therouter
```



Name	PID	Description	Status	Group
TermService		Remote Desktop Services	Stopped	NetworkService
Themes	1020	Themes	Running	netsvcs
THREADORDER		Thread Ordering Server	Stopped	LocalService
TrkWks	968	Distributed Link Tracking Client	Running	LocalSystemNetworkRestricted
TrustedInstaller	3228	Windows Modules Installer	Running	N/A
UI0Detect		Interactive Services Detection	Stopped	N/A
UMVPFSrv	352	UMVPFSrv	Running	N/A
upnphost		UPnP Device Host	Stopped	LocalServiceAndNoImpersonation
UxSms	968	Desktop Window Manager Session Manager	Running	LocalSystemNetworkRestricted
VaultSvc		Credential Manager	Stopped	
VBoxVmService	2204	VBoxVmService	Running	N/A
vds		Virtual Disk	Stopped	N/A
VSS		Volume Shadow Copy	Stopped	N/A
W32Time		Windows Time	Stopped	LocalService
WatAdminSvc		Windows Activation Technologies Service	Stopped	N/A
wbengine		Block Level Backup Engine Service	Stopped	N/A
WbioSrvc		Windows Biometric Service	Stopped	WbioSvcGroup
wcncsvc		Windows Connect Now - Config Registrar	Stopped	LocalServiceAndNoImpersonation
WcsPlugInService		Windows Color System	Stopped	wcssvc
WdiServiceHost	996	Diagnostic Service Host	Running	LocalService
WdiSystemHost	968	Diagnostic System Host	Running	LocalSystemNetworkRestricted
WebClient		WebClient	Stopped	LocalService
Wecsvc		Windows Event Collector	Stopped	NetworkService
wercplsupport		Problem Reports and Solutions Control Panel Support	Stopped	netsvcs

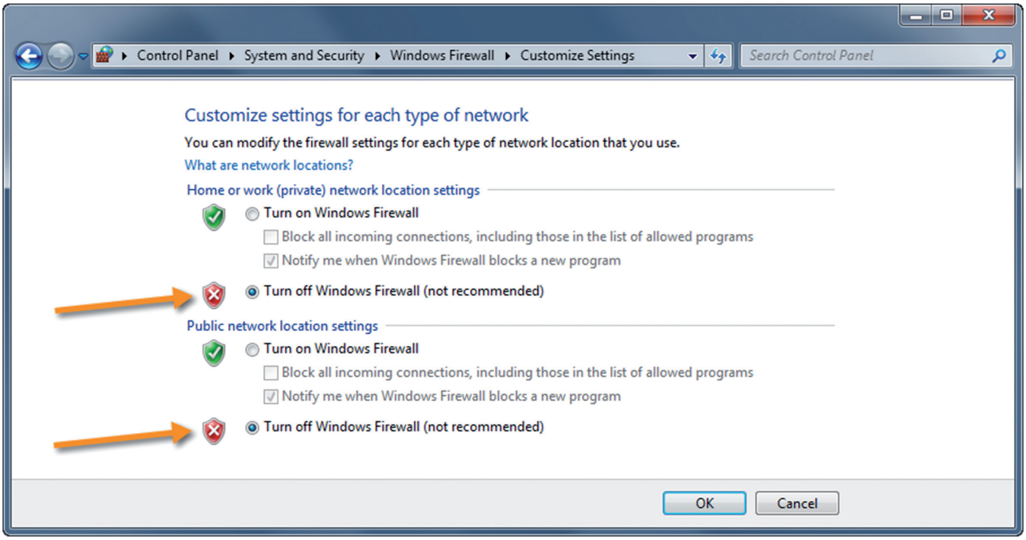
Η υπηρεσία VBoxVmService επιτρέπει τη λειτουργία ενός ή περισσοτέρων VirtualBox VMs ως services των Windows. Για την περίπτωση που εξετάζουμε, το OpenBSD VM θα ξεκινά αυτόματα κατά την εκκίνηση των Windows. Μεταξύ άλλων, αυτό σημαίνει πως πριν ακόμη κάνουμε login στον λογαριασμό ενός οποιουδήποτε χρήστη, το λειτουργικό σύστημα θα 'χει πρόσβαση στο δίκτυο.

Είναι φυσικά απαραίτητο να τροποποιήσουμε και το αρχείο /etc/dhcpd.conf, του DHCP server. Ξανά, δείτε το δικό μας:

```
option  domain-name "parabing.net";
option  domain-name-servers 192.168.200.1;

subnet 192.168.200.0 netmask 255.255.255.0 {
    option routers 8.8.8.8;
    range 192.168.200.100 192.168.200.150;
}
```

Ο καλύτερος τρόπος για να ληφθούν υπόψη οι αλλαγές είναι με μια επανεκκίνηση του λειτουργικού. Πριν όμως το κάνουμε, από το host OS εμφανίζουμε τον VirtualBox Manager, επιλέγουμε το VM με το OpenBSD και κάνουμε κλικ στο Settings. Στο παράθυρο ρυθμίσεων που θα εμφανιστεί πηγαίνουμε στην κατηγορία (καρτέλα) Network και συγκεκριμένα στο Adapter 2. Εκεί, φροντίζουμε ώστε για το «Attached to» να είναι επιλεγμένο το Bridged Adapter, καθώς και για το «Name» να είναι επιλεγμένο το Microsoft Loopback Adapter. Επικυρώνουμε τις αλλαγές με κλικ στο OK κι επανεκκινούμε το guest OS.



Κλείσιμο του Windows Firewall και για τα δύο είδη δικτύων που «ορίζουν» τα Windows (private και public). Από τη στιγμή που, πλέον, το λειτουργικό και οι εφαρμογές του προστατεύονται από το PF, το application firewall δεν είναι απαραίτητο.

Βήμα 7
Απενεργοποίηση TCP/IP για τον Ethernet adapter του PC

Στον Windows host, ακολουθούμε τη διαδρομή Control Panel → Network and Internet → Network and Sharing Center. Πάνω αριστερά υπάρχει ο σύνδεσμος Change adapter settings, όπου και κάνουμε ένα κλικ. Τα περιεχόμενα του παραθύρου θ' αλλάξουν και για κάθε network adapter, εικονικό (του VirtualBox) ή φυσικό (του συστήματος), θα δούμε ένα σχετικό εικονίδιο. Στις επιλογές προβολής του παραθύρου επιλέγουμε το Details και στρέφουμε την προσοχή μας στη στήλη Device Name. Σε μία από τις γραμμές της θα δούμε τον Microsoft Loopback Adapter και σε μία άλλη το μοντέλο του adapter που έως αυτή τη στιγμή χρησιμοποιεί το λειτουργικό, ώστε να βγαίνει στον έξω κόσμο. Κάνουμε κλικ πάνω στο όνομα της αντίστοιχης σύνδεσης (π.χ., Local Area Connection, βλ. στήλη Name) και στο νέο παράθυρο που θα εμφανιστεί (π.χ., με όνομα Local Area Connection Status) πατάμε στο Properties. Θα ανοίξει άλλο ένα παράθυρο, το Local Area Connection Properties. Παρατηρήστε το κεντρικό πλαίσιο. Εκεί, παρατίθενται τα πρωτόκολλα που είναι συσχετισμένα με το συγκεκριμένο network interface. Τα Internet Protocol Version 6 (TCP/IPv6) και Internet Protocol Version 4 (TCP/Ipv4) είναι τσεκαρισμένα. Καθαρίζουμε τα αντίστοιχα check boxes στ' αριστερά τους κι επικυρώνουμε τις αλλαγές με κλικ στο OK. Κλείνουμε και το παράθυρο Local Area Connection Status, με κλικ στο Close. Μετά από λίγα δευτερόλεπτα —και χωρίς να απαιτείται επανεκκίνηση του συστήματος—, για να βγαίνουν στον έξω κόσμο τα Windows θα χρησιμοποιούν τον Microsoft Loopback Adapter. Έτσι, όλο το network traffic θα περνά από το OpenBSD VM. Αν ανοίξουμε τώρα ένα παράθυρο γραμμής εντολών (Windows Command Processor) και πληκτρολογήσουμε εκεί

```
ipconfig
```

τότε μεταξύ των διαφόρων network adapters θα δούμε κι έναν με το όνομα Local Area Connection 2. Λογικά, το IP του θα πρέπει να προέρχεται από το διάστημα διευθύνσεων IP που μοιράζει ο DHCP server του OpenBSD. Στη δική μας περίπτωση, π.χ., ήταν το 192.168.200.100, δηλαδή η πρώτη διαθέσιμη από τις διευθύνσεις που έδινε ο DHCP.

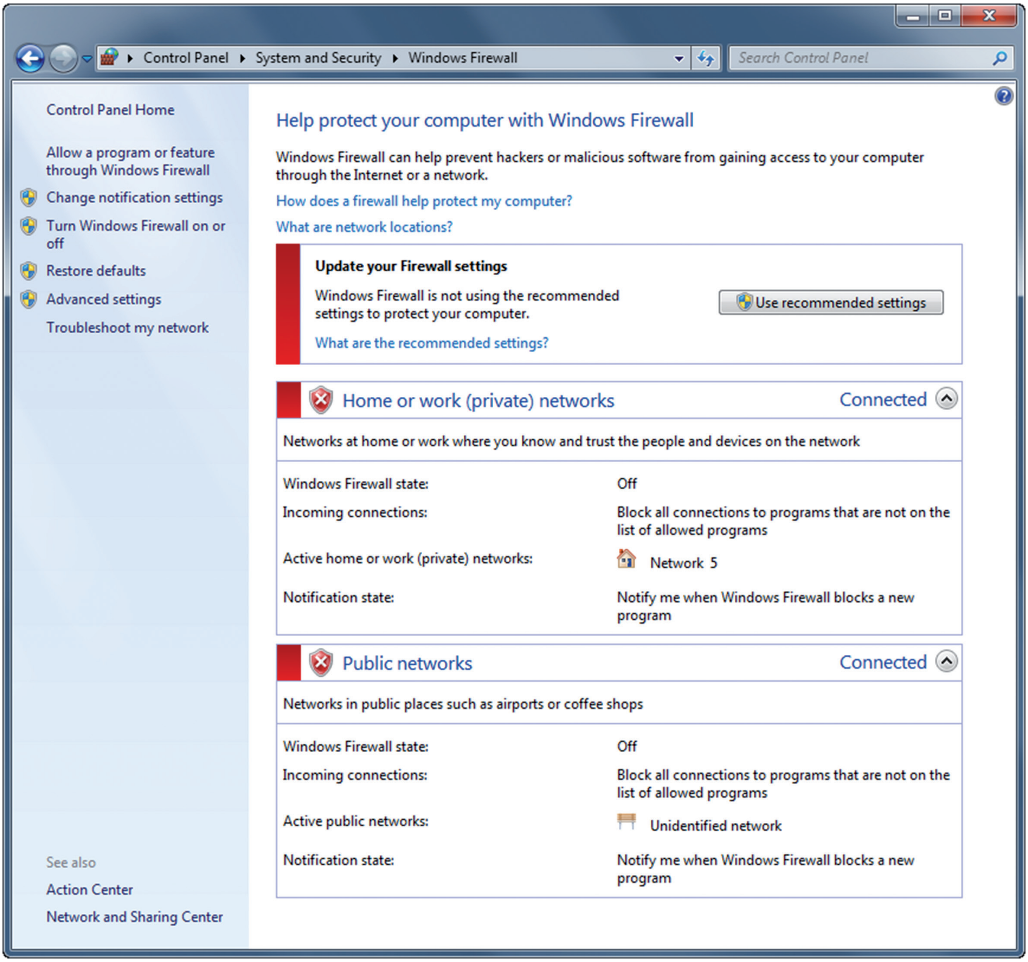
Βήμα 8
Δοκιμές με το host OS (Windows)

Πλέον, από τα Windows θα πρέπει να είμαστε σε θέση να κάνουμε ping στις διευθύνσεις των LAN και WAN adapters του OpenBSD VM, να έχουμε πρόσβαση σε υπηρεσίες άλλων μηχανημάτων του (αληθινού) τοπικού δικτύου (π.χ., σε κοινόχρηστους φεκέλους κι εκτυπωτές), καθώς και να βγαίνουμε στο Internet. Λόγω του NAT που γίνεται από το PF, η διεύθυνση IP που έχει το Windows PC στο τοπικό δίκτυο είναι αυτή που έχει ο WAN adapter του OpenBSD. Αν από κάποιον άλλον υπολογιστή του τοπικού δικτύου κάνουμε ένα port scan στη διεύθυνση αυτή, το *μόνο* port που θα βρούμε ανοικτό θα πρέπει να 'ναι το 22, του OpenSSH.

> **Σημείωση.** Στο εξής, όποτε θέλουμε να συνδεόμαστε στο OpenBSD VM μπορούμε να το κάνουμε μέσω SSH, π.χ., με τη βοήθεια του προγράμματος PuTTY (`http://bit.ly/getPuTTY`). Εξ ορισμού, ο OpenSSH server ακούει από το port 22 καθενός εκ των δύο network interfaces του OpenBSD.

Βήμα 9, προαιρετικό (αλλά κάντε το)
Απενεργοποίηση του application firewall

Δείχνουμε τη διαδικασία για το ενσωματωμένο Windows Firewall. Αν χρησιμοποιείτε κάποιο άλλο, ίσως θελήσετε να το απεγκαταστήσετε εντελώς. Πηγαίνουμε στο Control Panel κι ακολουθούμε τη διαδρομή System and Security → Windows Firewall → Turn Windows Firewall on or off (βλ. πάνω αριστερά). Τα περιεχόμενα του κεντρικού παραθύρου του Control Panel θα έχουν αλλάξει και θα δείτε δύο σετ ρυθμίσεων, για δύο είδη δικτύων: Home or work (private) network location settings και Public network location settings. Και για τα δύο είδη δικτύων επιλέγουμε το Turn off Windows Firewall (κι αγνοούμε την παρένθεση στα δεξιά, που γράφει «not recommended»). Επικυρώνουμε τις αλλαγές με ένα κλικ στο OK, κάτω δεξιά. Στο σημείο αυτό είναι καλή ιδέα να τροποποιήσουμε και τη συμπεριφορά του λεγόμενου Action Center, ώστε να μην παίρνουμε ειδοποιήσεις για το απενεργοποιημένο application firewall, για το πόσο κινδυνεύουμε χωρίς αυτό κ.λπ. κ.λπ. Για άλλη μια φορά, ανοίγουμε το Control Panel και δίνουμε System and Security → Action Center → Change Action Center settings (βλ. πάνω αριστερά). Παρατηρούμε τα check boxes κι αποεπιλέγουμε τα Internet security settings και Network firewall. Επικυρώνουμε με ένα κλικ στο OK, κάτω δεξιά. Ωραία. Ως εδώ απενεργοποιήσαμε το Windows Firewall, αποτρέψαμε και την εμφάνιση των σχετικών ειδοποιήσεων. Αν θέλουμε, μπορούμε να πάρουμε ακόμη πιο δραστικά μέτρα: Να απενεργοποιήσουμε και το σχετικό service του συστήματος. Για τελευταία φορά ανοίγουμε το Control Panel και δίνουμε System and Security → Administrative Tools. Εμφανίζεται ένα νέο παράθυρο και σ' αυτό κάνουμε κλικ στο Services. Αμέσως θα δούμε άλλο ένα παράθυρο, με το καθ' όλα αναμενόμενο όνομα «Services». Κάνουμε ένα κλικ στο όνομα της στήλης Name, ώστε να ταξινομήσουμε κατά όνομα υπηρεσίας. Εντοπίζουμε το Windows Firewall, κάνουμε δεξί κλικ πάνω του κι επιλέγουμε το Properties. Θα εμφανιστεί το παράθυρο ονόματι Windows Firewall Properties (Local Computer). Με ένα κλικ στο



Το Windows Firewall είναι «κατεβασμένο» και τα Windows είναι ...μέσα στο κόκκινο. Είναι που δεν έχουν ιδέα για την παρουσία του PF :D

Stop σταματάμε την υπηρεσία. Φροντίζουμε ώστε να μην ενεργοποιηθεί αυτόματα κατά την επόμενη εκκίνηση του λειτουργικού, θέτοντας Startup type = Disabled (βλ. drop-down menu, λίγο παραπάνω). Επικυρώνουμε με κλικ στο OK.

Βήμα 10
Κατέβασμα, εγκατάσταση και ρύθμιση VBoxVmService

Όπως είναι αυτή τη στιγμή το OpenBSD VM, μοιάζει λίγο πολύ με άλλη μία εφαρμογή των Windows. Κι επειδή αυτή η τελευταία πρόταση δεν λέει και πολλά, προσέξτε τι συμβαίνει:

- κάθε φορά που ξεκινούν τα Windows, πριν κάνουμε login σε κάποιον λογαριασμό το OpenBSD VM δεν είναι ενεργό
- ακόμη κι όταν κάνουμε login σε κάποιον λογαριασμό των Windows, το OpenBSD VM παραμένει ανενεργό

- για να ενεργοποιήσουμε το OpenBSD, πρέπει πρώτα να τρέξουμε τον VirtualBox Manager
- αφού τον τρέξουμε, επιλέγουμε το επιθυμητό VM, το ενεργοποιούμε και περιμένουμε να φορτώσει το guest OS — εν προκειμένω το OpenBSD

Όταν και μόνον όταν ολοκληρωθεί αυτό το τελευταίο βήμα, το host OS, δηλαδή τα Windows, αποκτά πρόσβαση στον έξω κόσμο. Προφανώς, η κατάσταση αυτή δεν είναι επιθυμητή. Ιδανικά, αυτό που θέλουμε είναι το OpenBSD VM να συμπεριφέρεται ως άλλο ένα Windows service. Οι (ενεργοποιημένες) υπηρεσίες των Windows ξεκινούν αυτόματα κατά την εκκίνηση του λειτουργικού συστήματος και είναι διαθέσιμες πριν καν συνδεθούμε στο λογαριασμό κάποιου χρήστη. Ειδικά όταν έχουμε να κάνουμε με μια κρίσιμη υπηρεσία, όπως είναι αυτή που μας επιτρέπει να `χουμε πρόσβαση στο Internet, καταλαβαίνετε ότι θέλουμε να ενεργοποιείται αυτόματα (κι όσο το δυνατόν νωρίτερα).

Από μόνο του, το VirtualBox δεν μας επιτρέπει να υποδεικνύουμε VMs τα οποία θέλουμε να συμπεριφέρονται ως services. Ευτυχώς, για τα Windows υπάρχει ένα εργαλείο που μας παρέχει αυτήν ακριβώς τη δυνατότητα. Πρόκειται για το VBoxVmService και μπορείτε να το κατεβάσετε από το

http://sourceforge.net/projects/vboxvmservice/files/vboxvmservice

Μπείτε στον κατάλογο με όνομα Versions 4.x και πάρτε το αρχείο

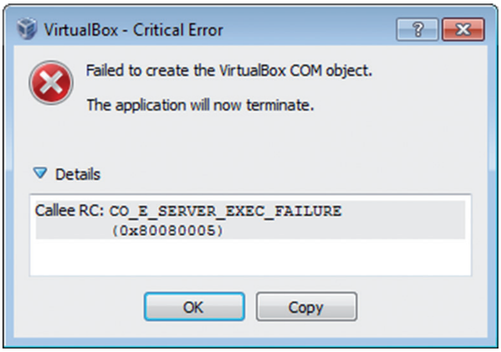
```
VBoxVmService-4.0-Bayberry.exe
```

Η εγκατάσταση δεν κρύβει κάποια έκπληξη. Εξ ορισμού, το εργαλείο πηγαίνει στον κατάλογο c:\vms. Αν θέλετε, εννοείται ότι μπορείτε να υποδείξετε κάποιον άλλον. Λίγο πριν ολοκληρωθεί η διαδικασία καλό είναι να αποεπιλέξουμε το κουτάκι για την ενεργοποίηση του VmServiceTray, αφού δεν το χρειαζόμαστε ακόμα. Αμέσως μετά πηγαίνουμε στον κατάλογο εγκατάστασης (π.χ., στον c:\vms). Με έναν text editor, όπως είναι το Notepad των Windows ή το Notepad++ (http://notepad-plus-plus.org), ανοίγουμε το αρχείο VBoxVmService.ini. Οι οδηγίες που περιλαμβάνονται μέσα στο αρχείο εν είδει σχολίων, αρκούν ώστε να γίνουν οι απαραίτητες τροποποιήσεις για την αυτόματη εκκίνηση του OpenBSD VM (ή και όποιου άλλου VM θέλουμε, δηλαδή). Δείτε το δικό μας VBoxVmService.ini:

```
[Settings]
VBOX_USER_HOME=C:\Users\
deltaHacker\.VirtualBox
RunWebService=no
PauseShutdown=5000

[Vm0]
VmName=Host firewall VM
ShutdownMethod=savestate
AutoStart=yes
```

Ουσιαστικά, στη δική σας περίπτωση αρκεί να αλλάξετε την τιμή της μεταβλητής VBOX_USER_HOME, βάζοντας στη



Όταν το VirtualBox λειτουργεί ως Windows service, η εκκίνηση του VirtualBox Manager είναι αδύνατη. Δεν χρειάζεται και πολλή σκέψη για να διαπιστώσουμε τι συμβαίνει. Εξάλλου το μήνυμα λάθους είναι σαφές: Failed to create the VirtualBox COM object :P

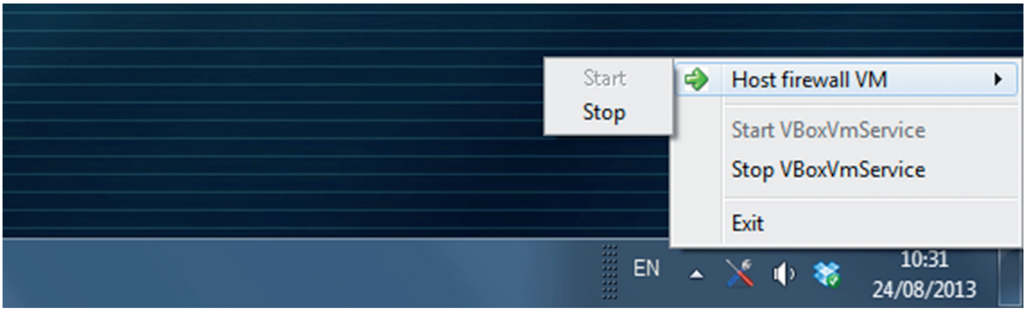
θέση του «deltaHacker» το κατάλληλο όνομα χρήστη. Επίσης, στη θέση αυτού του «Host firewall VM» θα δώσετε το όνομα του OpenBSD VM, ακριβώς όπως φαίνεται στον VirtualBox Manager. Αποθηκεύστε τις αλλαγές, κλείστε τον editor, κάντε ένα shutdown στο OpenBSD κι επανεκκινήστε τα Windows. Αν όλα έχουν πάει καλά, στο εξής το OpenBSD VM θα ξεκινά αυτόματα και δεν θα βλέπετε κανένα σχετικό παράθυρο στην οθόνη. Θα μπορείτε, φυσικά, να κάνετε SSH logins στο OpenBSD, π.χ., με τη βοήθεια του PuTTY. Επίσης, τα Windows θα έχουν κανονικότατη πρόσβαση στον έξω κόσμο, πάντα μέσω του OpenBSD και θα προστατεύονται από το PF.

Σημείωση. Αναλόγως της ταχύτητας του δίσκου στον οποίο βρίσκεται το OpenBSD VM, η εκκίνησή του ενδέχεται να καθυστερεί λίγο. Κατά συνέπεια, μπορεί π.χ. να έχετε προλάβει να συνδεθείτε στον λογαριασμό του χρήστη σας στα Windows, αλλά για λίγα δευτερόλεπτα να μην έχετε δίκτυο. Αφού πάντως ολοκληρώνεται η εκκίνηση του OpenBSD, όλα θα είναι και πάλι καλά. Προφανώς, αν το VM βρίσκεται σε δίσκο SSD τότε η εκκίνησή του θα γίνεται ταχύτατα και δεν θα παρατηρείτε τίποτε το μη-φυσιολογικό. Στο δικό μας PC *δεν* είχαμε δίσκο SSD, οπότε η καθυστέρηση που παρατηρούσαμε κατά την εκκίνηση των Windows ναι μεν δεν ήταν μηδαμινή, ωστόσο δεν ήταν και σπουδαία (μιλάμε για χρόνους της τάξης των 10 δευτερολέπτων).

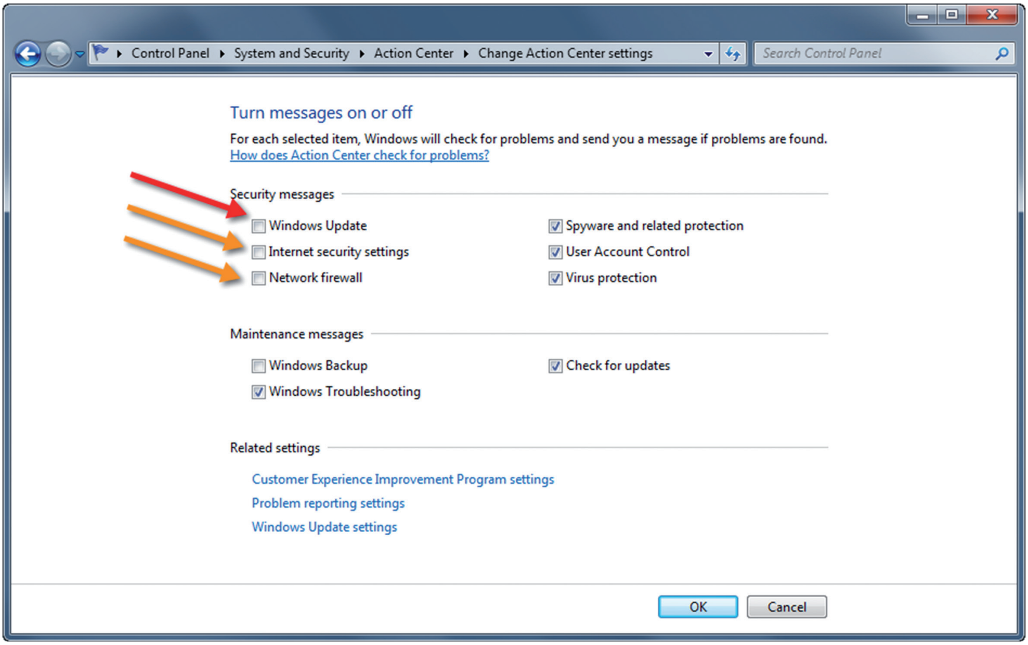
Βήμα 11, προαιρετικό
Ενεργοποίηση VirtualBox Manager και πρόσθετων VMs

Αν επιχειρήσετε να τρέξετε τον VirtualBox Manager, π.χ., επειδή θέλετε να σηκώσετε και κάποιο άλλο VM, το πρόγραμμα θα αποτύχει και αντ' αυτού θα δείτε στην οθόνη ένα όχι και τόσο ενθαρρυντικό μήνυμα λάθους. Αυτό συμβαίνει διότι όση ώρα το VirtualBox λειτουργεί ως service των Windows, η ταυτόχρονη λειτουργία και του VirtualBox Manager δεν είναι δυνατή. Τι κάνουμε, λοιπόν; Πολύ απλά, μπορούμε να τερματίσουμε προσωρινά την υπηρεσία του VirtualBox και μετά να ξεκινήσουμε τον Manager. Για την ακρίβεια, οι ενέργειες που κάνουμε είναι οι ακόλουθες.

- Αν το VmServiceTray δεν είναι παρόν στο system tray των Windows το καλούμε, π.χ., δίνοντας Windows Orb → All Programs → VBoxVmService → VmServiceTray.



Κάθε φορά που θέλουμε να τρέξουμε τον VirtualBox Manager —αν μη τι άλλο, προκειμένου να ξεκινήσουμε κι άλλα VMs— το εργαλειάκι VmServiceTray αποδεικνύεται πολύτιμο. Με τη βοήθειά του σταματάμε το OpenBSD VM (εμείς το έχουμε ονομάσει Host firewall VM) κι αμέσως μετά τερματίζουμε την υπηρεσία VBoxVmService (θα μας ζητηθεί επιβεβαίωση από το UAC). Σημειώστε ότι μετά από αυτές τις ενέργειες, τα Windows δεν έχουν πρόσβαση στο δίκτυο! Αλλά αυτό διορθώνεται εύκολα...



*Για να μη μας ζαλίζουν τα Windows διαρκώς με προειδοποιήσεις περί απουσίας (application) firewall κ.λπ., μια επίσκεψη στο Action Center κρίνεται απαραίτητη. Παρατηρήστε ότι, εμείς, έχουμε απενεργοποιήσει *και* τις ειδοποιήσεις που αφορούν στο Windows Update. Δεν σας προτείνουμε να μας μιμηθείτε — εκτός κι αν ξέρετε καλά τι κάνετε.*

- Με δεξί κλικ πάνω στο VmServiceTray εμφανίζεται ένα pop-up, στο οποίο θα δούμε *τουλάχιστον* το όνομα του OpenBSD VM (όπως το έχουμε ορίσει στο VBoxVmService.ini).
- Για εμάς, το όνομα αυτό είναι Host firewall VM κι αν πάμε το ποντίκι πάνω του εμφανίζεται ένα υπομενού με δύο εντολές: Start και Stop.
- Επιλέγουμε το Stop, κάνουμε ξανά διπλό κλικ πάνω από το VmServiceTray κι επιλέγουμε το Stop VBoxVmService. Προφανώς, αυτή τη στιγμή το OpenBSD VM είναι σταματημένο, η υπηρεσία του VirtualBox απενεργοποιημένη και το host OS δεν έχει πρόσβαση στο δίκτυο.
- Περιμένουμε λίγα δευτερόλεπτα και τρέχουμε τον VirtualBox Manager.
- Από τη λίστα με τα VMs επιλέγουμε πρώτα εκείνο με το OpenBSD (για εμάς, το Host firewall VM) και το ξεκινάμε με κλικ στο κουμπί Start. Θα παρατηρήσετε ότι ενεργοποιείται σχεδόν ακαριαία — κι αυτό διότι πριν από λίγο το VM δεν «κατέβηκε» με shutdown (βλ. οδηγία ShutdownMethod=savestate, στο VBoxVmService.ini).
- Τα Windows έχουν ξανά δικτύωση, από τον VirtualBox Manager μπορούμε να ξεκινάμε όποια κι όσα VMs θέλουμε (μνήμης επιτρέπουσας), ενώ έχουμε φυσικά τη δυνατότητα να φτιάχνουμε νέα VMs και γενικά να δουλεύουμε με το VirtualBox κανονικότατα.

Και μια τελευταία σημείωση: Αν κάθε VM που ενεργοποιούμε έχει το WAN interface του bridged με τον Microsoft Loopback Adapter, τότε θα βγαίνει στον έξω κόσμο μέσω του OpenBSD VM. Πάρα πολύ ωραία πράγματα, δηλαδή.

Βήμα 12
Μέριμνα ώστε το Windows PC να έχει σταθερό IP

Υπάρχουν λόγοι που θα θέλαμε το Windows PC να παίρνει πάντα το ίδιο IP από τον DHCP server του OpenBSD. Σκεφτείτε, π.χ., ότι χρειάζεται να ορίσουμε μερικούς κανόνες port forwarding στο PF, οι οποίοι αφορούν στο host OS. Θα δώσουμε σε λίγο συγκεκριμένο παράδειγμα, προς το παρόν όμως δείτε πώς πετυχαίνουμε το επιθυμητό αποτέλεσμα τροποποιώντας το αρχείο /etc/dhcpd.conf (στο OpenBSD):

```
option  domain-name "parabing.net";
option  domain-name-servers 8.8.8.8;

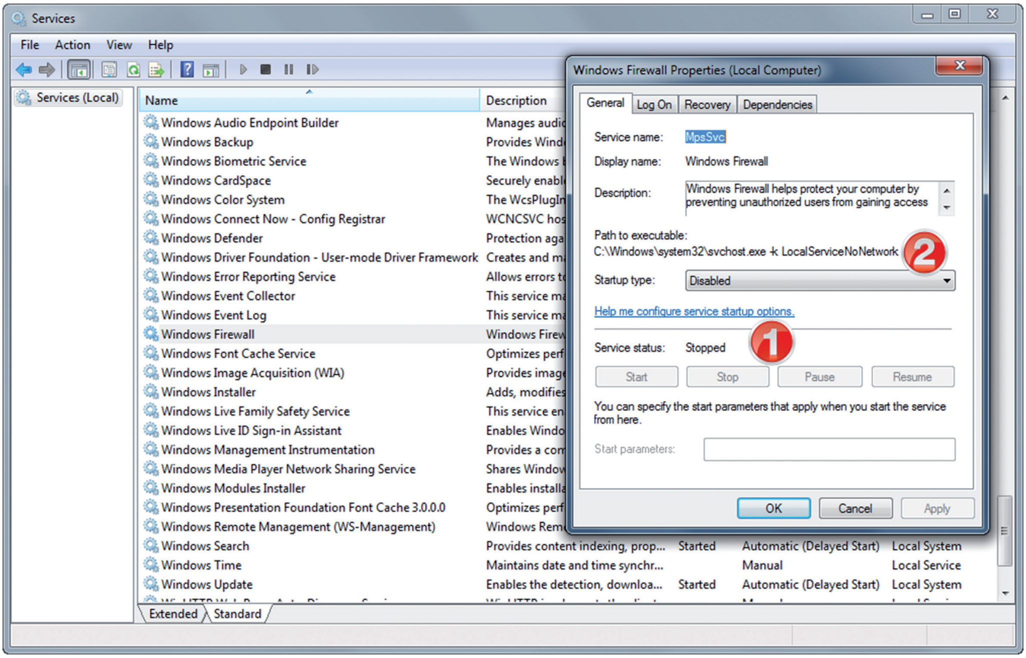
subnet 192.168.200.0 netmask 255.255.255.0 {
option routers 192.168.200.1;
range 192.168.200.100 192.168.200.150;

host windows-host {
hardware ethernet 02:00:4C:4F:4F:50;
fixed-address 192.168.200.10;
}
}
```

Παρατηρήστε τη νέα ενότητα «host ... {...}», την οποία έχουμε ενθέσει μέσα στην ενότητα «subnet ... {...}». Ουσιαστικά, πρόκειται για μια οδηγία που λέει κάτι σαν αυτό: «όταν ο DHCP client έχει το τάδε MAC address, τότε δώσε του ένα hostname και αντιστοίχισέ του πάντα το δείνα IP». Πρακτικά, λοιπόν, αυτό που οφείλουμε να βρούμε είναι το MAC address του Microsoft Loopback Adapter και ν' αποφασίσουμε ποια διεύθυνση IP θα παίρνει. Η προαναφερθείσα πρέπει να 'ναι εκτός του εύρους διευθύνσεων που μοιράζει ο DHCP — αλλά εντός του subnet στο οποίο λειτουργεί. Ένας τρόπος για να βρούμε τη διεύθυνση MAC του Loopback Adapter είναι με δεξί κλικ πάνω στο σχετικό εικονίδιο, επιλογή του Status και κλικ στο κουμπί Details. Στο παράθυρο που εμφανίζεται σημειώνουμε την τιμή που αντιστοιχεί στο πεδίο Physical Address (αυτή με τις παύλες). Να σημειώσουμε, εξάλλου, ότι αφού γίνουν οι απαραίτητες αλλαγές στο /etc/dhcpd.conf του OpenBSD, προκειμένου να ληφθούν υπόψη αρκεί να επανεκκινήσουμε την υπηρεσία με ένα

```
# /etc/rc.d/dhcpd restart
```

Ακολούθως απενεργοποιούμε κι αμέσως ενεργοποιούμε ξανά τον Loopback Adapter ή, εναλλακτικά, κάνουμε ένα restart στα Windows. Πριν συνεχίσουμε, επαληθεύουμε ότι πράγματι έχουμε πάρει το επιθυμητό, στατικό IP.



*Η τελευταία γέφυρα με το Windows Firewall καίγεται απενεργοποιώντας την αντίστοιχη υπηρεσία (Windows service). Παρατηρήστε ότι πρώτα τη σταματάμε (1) και μετά φροντίζουμε ώστε να *μην* ενεργοποιείται αυτόματα, κατά την εκκίνηση των Windows (2).*

Βήμα 13
Port forwarding για το filesharing των Windows

Αν ο υπολογιστής με τα Windows δεν είναι laptop, οπότε βρίσκεται όλη την ώρα στο ίδιο τοπικό δίκτυο, τότε βολεύει να έχει την υπηρεσία file & print sharing ενεργοποιημένη. Προκειμένου άλλοι υπολογιστές να έχουν πρόσβαση σε κοινόχρηστους φακέλους κι εκτυπωτές των Windows, χρησιμοποιούνται τέσσερα (4) ανοικτά ports: 139/TCP, 445/TCP, 137/UDP και 138/UDP. Μιας και τα Windows δεν έχουν πλέον κάποιο application firewall, όλα τα ports τους είναι ανοικτά. Δεν ισχύει όμως το ίδιο για το PF, μπροστά από τα Windows. Για την ακρίβεια, προς το παρόν το PF επιτρέπει *μόνο* τα εισερχόμενα πακέτα που κατευθύνονται στο port 22/TCP του WAN interface (em0). Εκείνο που χρειάζεται, λοιπόν, είναι να τροποποιήσουμε το αρχείο /etc/pf.conf ώστε το PF...

α) να επιτρέπει τα εισερχόμενα πακέτα προς τα ports 139/TCP, 445/TCP, 137/UDP και 138/UDP,

β) να τα ανακατευθύνει στα αντίστοιχα ports του Windows box, πίσω του.

Πριν λίγο φροντίσαμε ώστε τα Windows να παίρνουν πάντα το ίδιο IP από τον DHCP server του OpenBSD, οπότε τώρα θα μπορέσουμε να ορίσουμε εύκολα τους κατάλληλους κανόνες port forwarding, ώστε όλοι οι υπολογιστές του (φυσικού) τοπικού δικτύου να έχουν πρόσβαση στους κοινόχρηστους φακέλους κι εκτυπωτές των Windows. Δείτε το δικό μας pf.conf και τροποποιήστε το δικό σας αναλόγως.

```
wan_if="em0"
lan_if="em1"
```

```
allowed_icmp_types="echoreq"
allowed_tcp_service_ports="{ 22 }"

filesharing_tcp_ports="{ 139, 445 }"
filesharing_udp_ports="{ 137, 138 }"

thehost="192.168.200.10"
set block-policy drop
set loginterface $wan_if

set skip on lo

match out on $wan_if inet from !($wan_if:network) to any nat-to ($wan_if:0)

block in log
pass out quick

antispoof quick for { lo $lan_if }

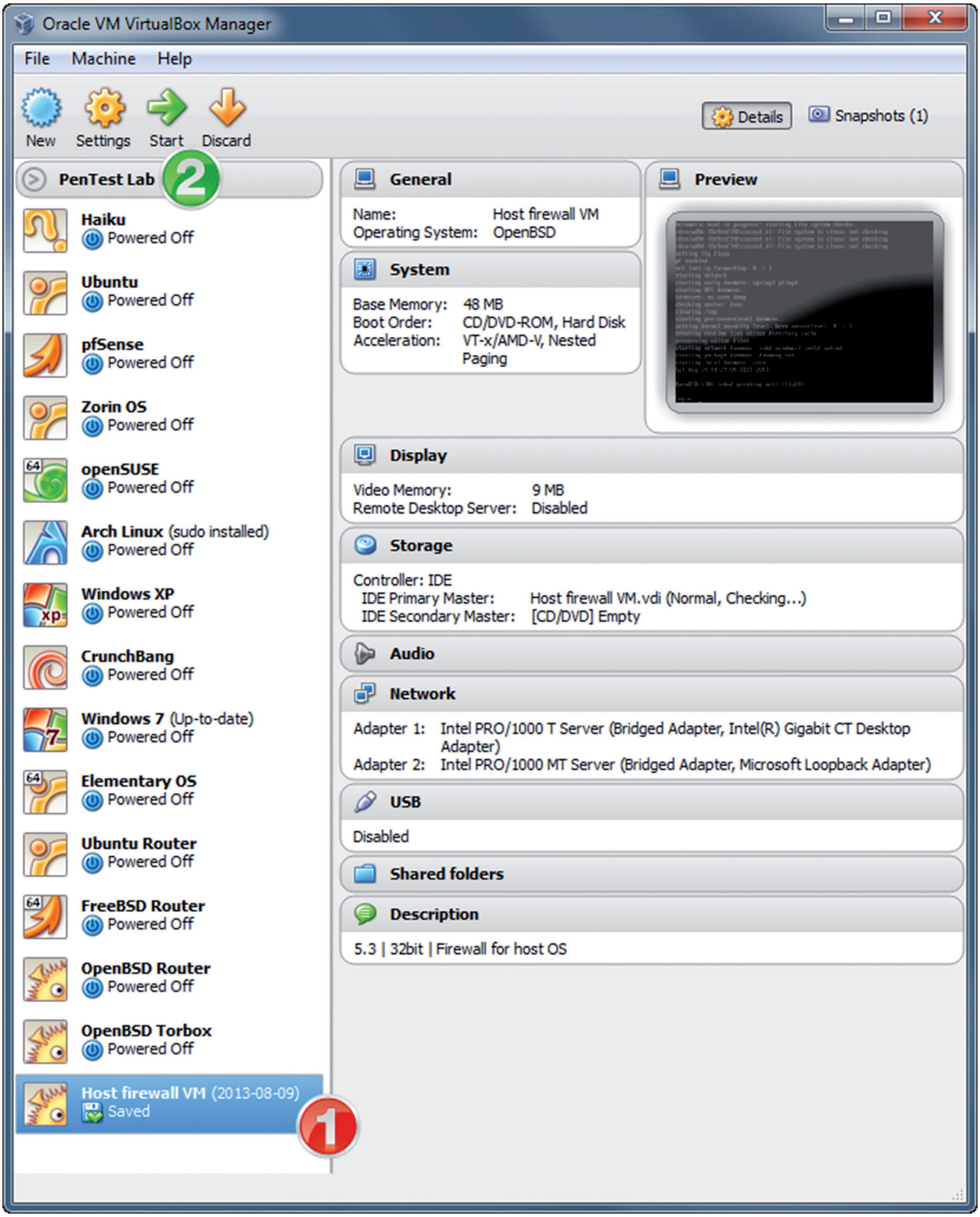
pass in on $wan_if inet proto tcp from any \
    to ($wan_if) port $allowed_tcp_service_ports

pass in on $wan_if inet proto tcp from $wan_if:network \
    to ($wan_if) port $filesharing_tcp_ports rdr-to $thehost
pass in on $wan_if inet proto udp from $wan_if:network \
    to ($wan_if) port $filesharing_udp_ports rdr-to $thehost
pass in inet proto icmp all icmp-type $allowed_icmp_types

pass in on $lan_if
```

Παρατηρήστε τις λίστες filesharing_tcp_ports, filesharing_udp_ports, καθώς και το macro thehost. Δώσαμε περιγραφικές ονομασίες διότι θέλουμε, όσος καιρός κι αν περάσει, να ρίχνουμε μια ματιά στο αρχείο pf.conf κι αμέσως να καταλαβαίνουμε τι κάνουν οι κανόνες. Γενικά, το εν λόγω αρχείο προτείνεται να το κάνουμε όσο πιο human readable γίνεται — αποφεύγοντας φυσικά τις υπερβολές. Δείτε τώρα πόσο εύκολα διαβάζονται οι δύο κανόνες για το port forwarding, προς το τέλος του pf.conf. (Η οδηγία rdr-to σημαίνει «redirect to», δηλαδή «ανακατεύθυνε σε».) Αφού αποθηκεύσουμε τις αλλαγές στο pf.conf, φορτώνουμε στο PF το νέο ruleset:

```
# pfctl -f /etc/pf.conf
```



Μετά τον τερματισμό της υπηρεσίας VBoxVmService, η εκκίνηση του VirtualBox Manager είναι και πάλι δυνατή. Στη λίστα με τα VMs, το πρώτο πράγμα που κάνουμε είναι να επιλέξουμε εκείνο το VM σε ρόλο router/firewall για το host computer (1) κι αμέσως να το ξεκινήσουμε με ένα κλικ στο Start (2). Έτσι, τα Windows θα έχουν και πάλι πρόσβαση στο δίκτυο. Επιπρόσθετα, μπορούμε πλέον να ξεκινήσουμε όποια κι όσα άλλα VMs θέλουμε. Παρεμπιπτόντως, αν το επιθυμούμε κάποια απ' αυτά θα βγαίνουν στον έξω κόσμο μέσω του OpenBSD VM!

Βήμα 14
Στατικό IP για τον WAN adapter του OpenBSD

Αν το host computer δεν μετακινείται από δίκτυο σε δίκτυο, όπως, π.χ., συμβαίνει με τα laptops, τότε το WAN interface του OpenBSD, άρα και το δημόσιο IP των Windows, βολεύει να είναι στατικό. Έτσι, η πρόσβαση σε υπηρεσίες file & print sharing των Windows από άλλα μηχανήματα του (φυσικού) τοπικού δικτύου γίνεται πιο εύκολα και άμεσα — αρκεί βέβαια να θυμόμαστε τη διεύθυνση IP του WAN interface. Επίσης, στον φυσικό router μπροστά από το Windows box είναι πλέον εφικτό να ορίζουμε κανόνες port forwarding για το OpenBSD, ώστε, π.χ., να έχουμε απομακρυσμένη (δηλαδή από το Internet) πρόσβαση στο λειτουργικό, μέσω SSH.

Στο δικό μας τοπικό δίκτυο, μέσα στο οποίο κάναμε τις δοκιμές μας, οι συσκευές παίρνουν IPs της μορφής 10.20.30.*. Αποφασίσαμε να δώσουμε στο OpenBSD/Windows τη διεύθυνση 10.20.30.10 — και δείτε πώς το πετύχαμε.

Για το OpenBSD, το device name του WAN interface είναι το em0 και πριν την αλλαγή τα περιεχόμενα του αρχείου /etc/hostname.em0 έχουν μία μόνο λέξη:

```
dhcp
```

Ανοίγουμε το αρχείο με έναν text editor και φροντίζουμε ώστε να έχει μία μόνο γραμμή:

```
inet 10.20.30.10 255.255.255.0
```

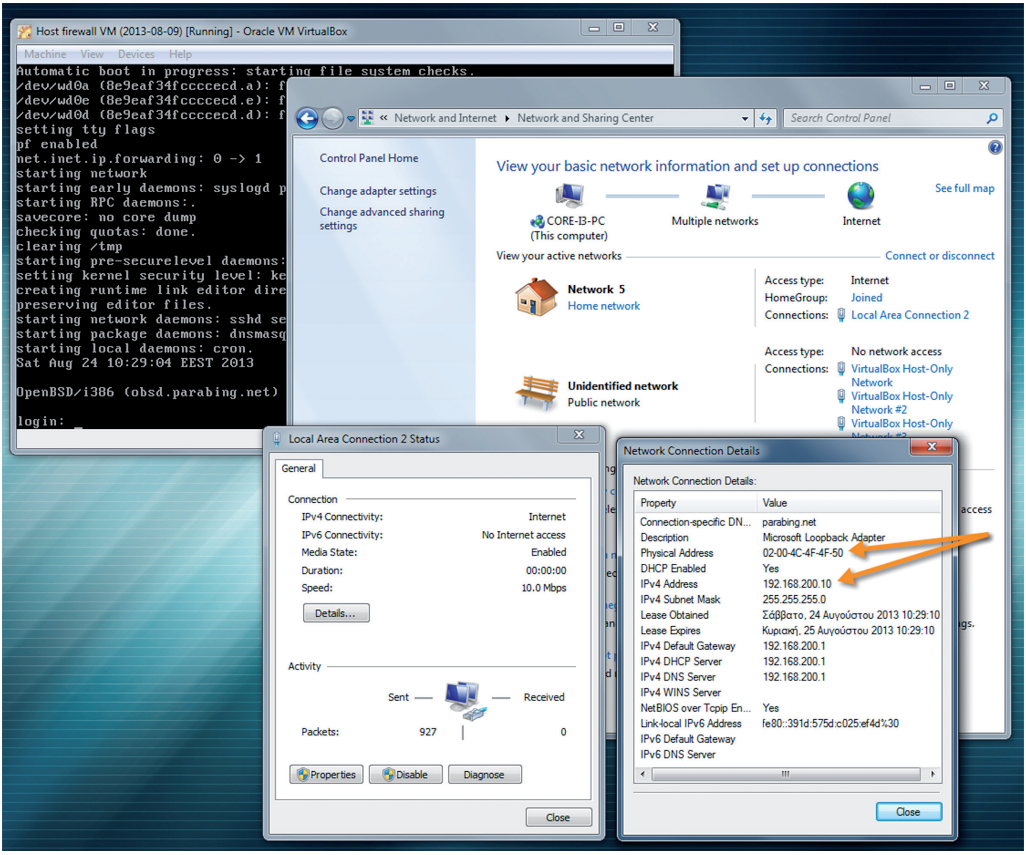
Αποθηκεύουμε την αλλαγή και μετά ενημερώνουμε κατάλληλα και το αρχείο /etc/hosts. Παράδειγμα:

```
10.20.30.10     obsd.parabing.net obsd
```

Σειρά έχει τώρα το αρχείο /etc/resolv.conf, το οποίο πρέπει να περιλαμβάνει τη διεύθυνση IP τουλάχιστον ενός name server. Μέχρι στιγμής, τα περιεχόμενα του εν λόγω αρχείου ενημερώνονταν από τον DHCP client του OpenBSD. Πλέον, όμως, κανένα από τα network interfaces του λειτουργικού δεν θα παίρνει δυναμικά IP από κάποιον DHCP server, οπότε το σύστημα δεν θα λαμβάνει και συνοδευτικές πληροφορίες, όπως, π.χ., διευθύνσεις IP για name servers. Μπορούμε να επιλέξουμε κάποιον δημόσιο name server — για παράδειγμα έναν από τους δημοφιλέστατους της Google (8.8.8.8). Εναλλακτικά, έχουμε την επιλογή της διεύθυνσης IP που διαθέτει ο router του (αληθινού) τοπικού δικτύου (για εμάς 10.20.30.1). Όπως και να 'χει, οφείλουμε να τροποποιήσουμε το αρχείο /etc/resolv.conf κατάλληλα:

```
# Generated by em0 dhclient
search home
nameserver 8.8.8.8
lookup file bind
```

(Αγνοήστε την πρώτη γραμμή, που είχε προστεθεί από τον DHCP client του OpenBSD. Αγνοήστε κι αυτό το «home» στη δεύτερη γραμμή, το οποίο είναι το προκαθορισμένο όνομα domain του VDSL router μας.) Μένει να φτιάξουμε κι ένα νέο αρχείο, το /etc/mygate, το οποίο θα πρέπει να περιλαμβάνει μία και μόνο μία διεύθυνση IP: αυτή του gateway. Πρόκειται για την εσωτερική διεύθυνση IP που έχει ο router μπροστά από το Windows box και για τη δική μας περίπτωση είναι η 10.20.30.1. Για τον δικό σας router, η συζητούμενη διεύθυνση IP μπορεί να 'ναι 192.168.0.1, 192.168.1.1 ή



Είναι καλή ιδέα να ρυθμίσουμε κατάλληλα τον DHCP server στο OpenBSD VM, ώστε το Windows host να παίρνει πάντα την ίδια διεύθυνση IP. Θα μπορούμε έτσι να ορίζουμε εύκολα κανόνες port forwarding στο PF, οι οποίοι θα αφορούν στο host OS (Windows). Προκειμένου να κάνουμε τη σχετική ρύθμιση στον DHCP server, είναι απαραίτητο να σημειώσουμε κάπου το MAC address του Microsoft Loopback Adapter.

ακόμη και μια πιο «παράξενη», όπως 10.0.0.1, 10.0.0.254 κ.ο.κ.

Η δημιουργία του αρχείου /etc/mygate και το ταυτόχρονο άνοιγμά του για editing γίνονται με μια μόλις εντολή, π.χ., κάπως έτσι:

```
# nano /etc/mygate
```

(Το nano είναι ο αγαπημένος σας text editor στο Unix, έτσι δεν είναι; :D) Το δικό μας /etc/mygate έχει μία μόνο γραμμή:

```
10.20.30.1
```

Και μετά από όλες τις απαραίτητες παρεμβάσεις, το WAN interface του OpenBSD είναι έτοιμο να πάρει στατική διεύθυνση IP. Για να βεβαιωθούμε ότι όλα έχουν πάει καλά, προτείνεται να επανεκκινήσουμε με ένα

```
# reboot
```

Αφού σηκωθεί ξανά το σύστημα, συνδεόμαστε ξανά στον λογαριασμό του root (π.χ., με SSH login στη διεύθυνση του LAN interface) κι επαληθεύουμε ότι όλα είναι όπως θέλουμε:

```
# ifconfig
[...]
em0: flags=8843<UP,BROADCAST,RUNNING,SIMPLEX,MULTICAST> mtu 1500
        lladdr 08:00:27:3a:78:a2
        priority: 0
        groups: egress
        media: Ethernet autoselect (1000baseT full-duplex)
        status: active
        inet 10.20.30.10 netmask 0xffffff00 broadcast 10.20.30.255
        inet6 fe80::a00:27ff:fe3a:78a2%em0 prefixlen 64 scopeid 0x1
em1: flags=8843<UP,BROADCAST,RUNNING,SIMPLEX,MULTICAST> mtu 1500
        lladdr 08:00:27:05:a3:2a
        priority: 0
        media: Ethernet autoselect (1000baseT full-duplex)
        status: active
        inet 192.168.200.1 netmask 0xffffff00 broadcast 192.168.200.255
        inet6 fe80::a00:27ff:fe05:a32a%em1 prefixlen 64 scopeid 0x2
[...]
```

Βήμα 15, προαιρετικό
Tor middlebox για *κάποιους* πελάτες

Στο deltaHacker 022 δείχνουμε πώς φτιάχνουμε ένα Tor middlebox με βάση το OpenBSD. Ουσιαστικά, πρόκειται για έναν router ο οποίος περνάει όλη ανεξαιρέτως τη δικτυακή κίνηση των πελατών του μέσα από το δίκτυο ανωνυμίας του Tor. Τις λεπτομέρειες που αφορούν στη δημιουργία του μπορείτε να διαβάσετε στο άρθρο που ξεκινά από τη σελίδα 74 του τεύχους 022. Στο εν λόγω άρθρο, από το δίκτυο του Tor περνάει η κίνηση όλων των υπολογιστών που εξυπηρετούνται από το OpenBSD. Σε ένα πιο ρεαλιστικό σενάριο, από το Tor θα θέλουμε να περνά η κίνηση μερικών μόνον από τους υπολογιστές «πίσω» από το OpenBSD. Στην περίπτωση που εξετάζουμε τώρα, π.χ., ίσως αποφασίζαμε να μην περνά από το Tor network η κίνηση του Windows host, αλλά να περνά η κίνηση κάθε VM που χρησιμοποιεί το OpenBSD VM ως router. Αυτό μπορούμε να το πετύχουμε τροποποιώντας κατάλληλα το αρχείο /etc/pf.conf. Δείτε:

```
wan_if="em0"
lan_if="em1"
```

```
allowed_icmp_types="echoreq"
allowed_tcp_service_ports="{ 22 }"

filesharing_tcp_ports="{ 139, 445 }"
filesharing_udp_ports="{ 137, 138 }"

thehost="192.168.200.10"

table <torrified> { $lan_if:network, ! $thehost }

set block-policy drop
set loginterface $wan_if
set skip on lo

match out on $wan_if inet from !($wan_if:network) to any nat-to ($wan_if:0)

block in log
pass out quick

antispoof quick for { lo $lan_if }

pass in on $wan_if inet proto tcp from any \
    to ($wan_if) port $allowed_tcp_service_ports

pass in on $wan_if inet proto tcp from $wan_if:network \
    to ($wan_if) port $filesharing_tcp_ports rdr-to $thehost
pass in on $wan_if inet proto udp from $wan_if:network \
    to ($wan_if) port $filesharing_udp_ports rdr-to $thehost

pass in inet proto icmp all icmp-type $allowed_icmp_types

pass in on $lan_if

pass in quick on $lan_if inet proto tcp from <torrified> \
    to !($lan_if:network) rdr-to 127.0.0.1 port 9040
pass in quick on $lan_if inet proto udp \
    to port 53 rdr-to 127.0.0.1 port 54
```

Το κλειδί εδώ είναι ο πίνακας ονόματι torrified, τον οποίο εισάγουμε ώστε να αποθηκεύει τις διευθύνσεις IP των υπολογιστών, η κίνηση των οποίων *θέλουμε* να περνάει από το δίκτυο του Tor. Ο εν λόγω πίνακας, λοιπόν, περιλαμβάνει όλες τις έγκυρες διευθύνσεις IP του δικτύου πίσω από το LAN interface του OpenBSD, εκτός όμως από τη διεύθυνση του host computer. Η προτελευταία γραμμή του pf.conf, λέει το εξής: Τα πακέτα που φτάνουν στο LAN interface από οποιονδήποτε υπολογιστή του τοπικού δικτύου *εκτός* από τον host και προορίζονται για τον έξω κόσμο, ανακατεύθυνέ τα στο port 9040 του τοπικού tor proxy (πεζό «t»). Έτσι, η δικτυακή κίνηση όλων των VMs είναι torrified, εκτός από την κίνηση του Windows host. Προσέξτε τώρα την τελευταία γραμμή του pf.conf: Επειδή θέλουμε *και* τα DNS queries να είναι ανώνυμα, τα περνάμε κι αυτά από τον τοπικό tor proxy. Αλλά τι γίνεται με τα σχετικά queries του Windows host; Γίνεται ότι *και* αυτά εξυπηρετούνται δια μέσω του Tor network. Το πρόβλημα όμως είναι ότι έτσι εισάγονται περιττές καθυστερήσεις, για ένα μηχάνημα του οποίου η κίνηση ούτως ή άλλως δεν είναι ανώνυμη. Μπορούμε να κάνουμε κάτι γι' αυτό; Συνεχίστε το διάβασμα.

Βήμα 16, προαιρετικό
Nameserver για *κάποιους* πελάτες

Θα εγκαταστήσουμε τον dnsmasq, έναν DNS forwarder για όσους υπολογιστές βγαίνουν στον έξω κόσμο απευθείας κι όχι μέσω του Tor network. Επειδή ο dnsmasq έχει *και* δυνατότητες DHCP, ως άσκηση θα απενεργοποιήσουμε τον υπάρχοντα DHCP server και θα συνεχίσουμε μ' εκείνον του dnsmasq. Προσέξτε: Αυτό δεν είναι υποχρεωτικό, όμως σκεφτήκαμε να το κάνουμε ώστε αφενός να μάθουμε καλύτερα το dnsmasq και τις δυνατότητές του, αφετέρου επειδή θέλαμε όσο το δυνατόν λιγότερους ενεργούς servers στο ωραιότατο OpenBSD VM μας.

Ξεκινάμε με την εγκατάσταση του dnsmasq:

```
# pkg_add -v dnsmasq
```

Δημιουργούμε ένα αντίγραφο εφεδρείας του πρωτότυπου αρχείου ρυθμίσεων:

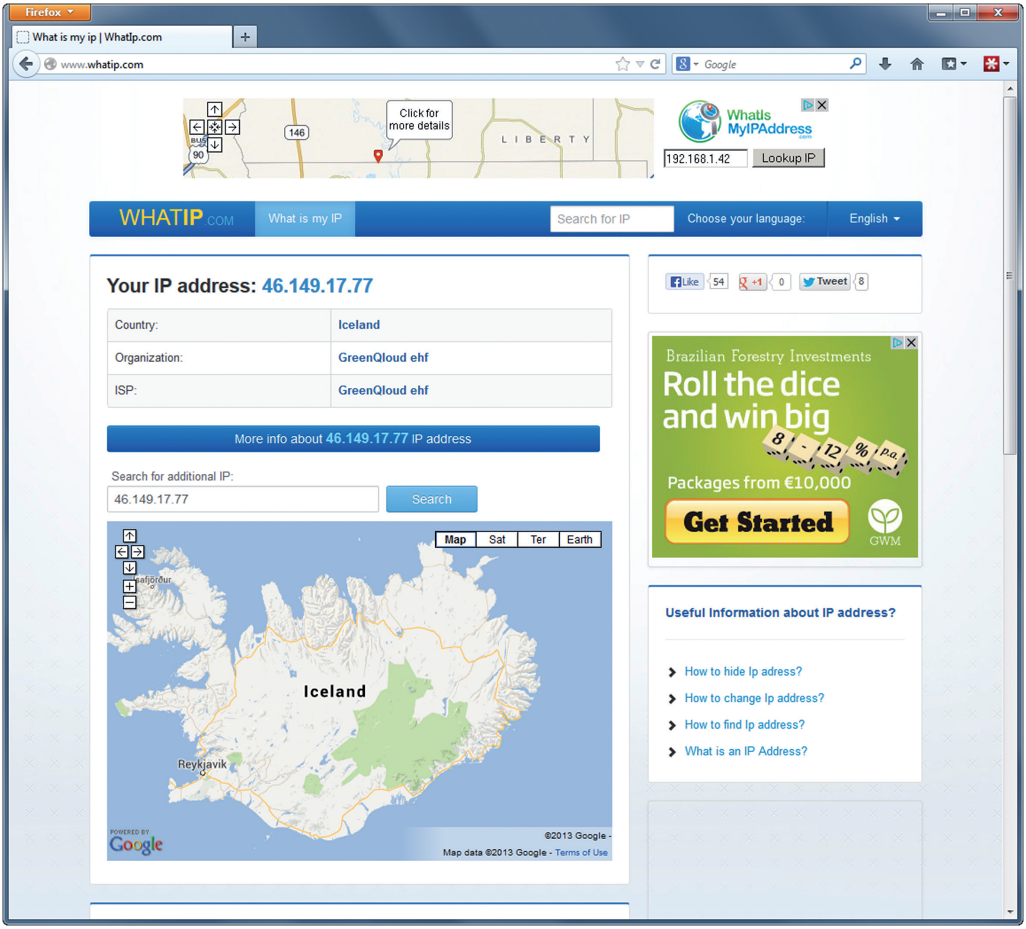
```
# cp /etc/dnsmasq.conf /etc/dnsmasq.conf.original
```

Με έναν text editor όπως το nano, ανοίγουμε το dnsmasq.conf προς διόρθωση:

```
# nano /etc/dnsmasq.conf
```

Τα σχόλια στο αρχείο είναι αρκετά επεξηγηματικά. Παραθέτουμε εδώ τις γραμμές που τροποποιήσαμε εμείς. Κοιτάζοντας κάθε μία εξ αυτών, προτείνουμε ν' αναζητήσετε την αντίστοιχη γραμμή στο /etc/dnsmasq.conf και να την τροποποιήσετε αναλόγως. Σε κάποιες περιπτώσεις θα αρκεί απλά να την «αποσχολιάσετε», δηλαδή να διαγράψετε τον χαρακτήρα «#» στ' αριστερά της, ώστε να μη θεωρείται ως σχόλιο κι έτσι να λαμβάνεται υπόψη. Σε άλλες περιπτώσεις θα κάνετε λίγες περισσότερες επεμβάσεις, π.χ., για να ορίσετε το εύρος διευθύνσεων που μοιράζει ο DHCP του dnsmasq.

```
port=53
domain-needed
bogus-priv
filterwin2k
domain=parabing.net
```



Αφού προσθέσαμε στο OpenBSD VM και τις δυνατότητες ενός Tor middlebox, αρχίσαμε τις δοκιμές. Πολύ σύντομα βγήκαμε στο Internet από exit node που βρίσκεται σ' ένα από τα πιο αγαπημένα μας νησάκια στον πλανήτη :D

```
6| dhcp-range=192.168.200.100,192.168.200.150,12h
7| dhcp-host=02:00:4C:4F:4F:50,win7host,192.168.200.10,24h
8| dhcp-option=252,"\n"
```

Σημειώστε ότι οι αριθμοί γραμμών δεν αποτελούν μέρος του πρωτότυπου αρχείου και τους προσθέσαμε για ευκολία αναφοράς. Στη γραμμή 1, προσέξτε το port από το οποίο ακούει ο nameserver: είναι το προκαθορισμένο, το 53. Ο nameserver του tor proxy, από την άλλη, είναι ρυθμισμένος για ν' ακούει από το port 54. Στη γραμμή 6, ορίζουμε το εύρος διευθύνσεων που μοιράζει ο DHCP, καθώς και το χρόνο του lease (12 ώρες). Τέλος, στη γραμμή 7 φροντίζουμε ώστε το Windows host να παίρνει πάντα το ίδιο IP.

Αποθηκεύουμε τις αλλαγές στο /etc/dnsmasq.conf κι ανοίγουμε το αρχείο /etc/rc.conf.local. Εκεί, σχολιάζουμε τη γραμμή που είναι υπεύθυνη για την ενεργοποίηση του DHCP server κατά την εκκίνηση του OpenBSD. Τροποποιούμε επίσης την τιμή

της μεταβλητής pkg_scripts, ώστε πριν από τον tor proxy να ξεκινά αυτόματα κι ο dnsmasq. Δείτε τα περιεχόμενα του αρχείου, μετά τις τροποποιήσεις:

```
#dhcpd_flags="em1"
pkg_scripts="dnsmasq tor"
```

Πριν απενεργοποιήσουμε χειροκίνητα τον DHCP server κι ενεργοποιήσουμε —επίσης χειροκίνητα— τον dnsmasq, κάνουμε μια μόνο τροποποίηση στο αρχείο /etc/pf.conf. Πιο συγκεκριμένα, η τελευταία του γραμμή ήταν η

```
pass in quick on $lan_if inet proto udp \
to port 53 rdr-to 127.0.0.1 port 54
```

και τώρα την αλλάζουμε ώστε να είναι σαν αυτή:

```
pass in quick on $lan_if inet proto udp \
from <torrified> to port 53 rdr-to 127.0.0.1 port 54
```

Είναι, δηλαδή, σαν να λέμε ότι μόνο τα DNS queries των torrified υπολογιστών/VMs θα εξυπηρετούνται από τον DNS του tor proxy. Τα αντίστοιχα queries των υπόλοιπων υπολογιστών/VMs θα φτάνουν στο port 53, στο LAN interface του OpenBSD, και θα τα διαχειρίζεται ο dnsmasq. Απενεργοποιούμε τον DHCP server

```
# /etc/rc.d/dhcpd stop
```

ενεργοποιούμε τον dnsmasq

```
# /etc/rc.d/dnsmasq start
```

και φορτώνουμε στο PF το νέο ruleset:

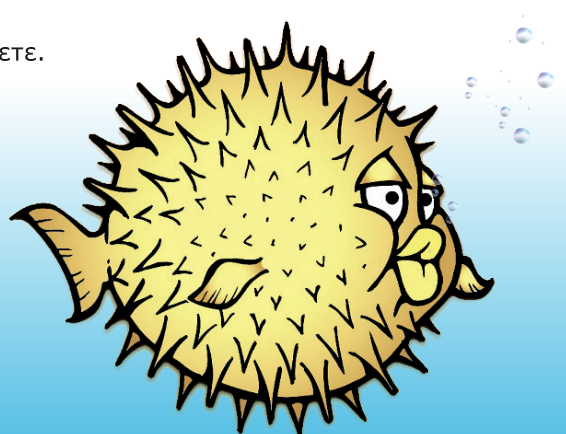
```
# pfctl -f /etc/pf.conf
```

Τι καταφέραμε;

Αν φτάσατε έως εδώ κι έχετε διαβάσει ή/και εφαρμόσει όλα τα προηγούμενα, τότε η ερώτηση είναι άκυρη. Αν πάλι ανατρέξατε σ’ αυτή την παράγραφο κατ’ ευθείαν, τότε θα σας πούμε ότι εκτός από ένα πανίσχυρο κι εξαιρετικά ασφαλές firewall για το Windows box σας, παίζοντας με τα OpenBSD, PF αλλά και με υπηρεσίες όπως το dnsmasq, αποκτάτε πολύτιμες γνώσεις που λίγοι έχουν.

Δεν χρειάζεται να μας πιστέψετε.

Αργά ή γρήγορα θα το διαπιστώσετε.



deltahacker.gr